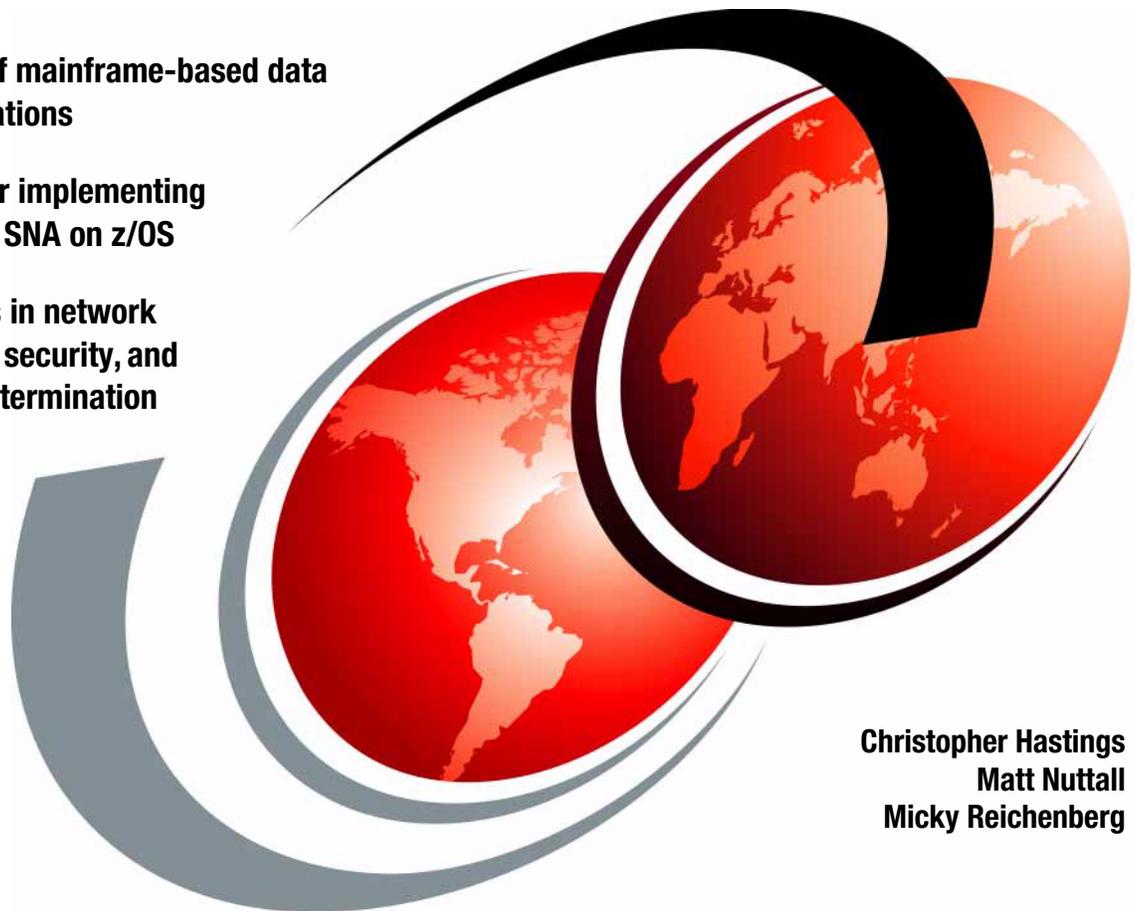


# Introduction to the New Mainframe: Networking

Concepts of mainframe-based data communications

Methods for implementing TCP/IP and SNA on z/OS

Basic skills in network operations, security, and problem determination



Christopher Hastings  
Matt Nuttall  
Micky Reichenberg





International Technical Support Organization

**Introduction to the New Mainframe: Networking**

August 2006

**Note:** Before using this information and the product it supports, read the information in “Notices” on page 329.

**First Edition (August 2006)**

This edition applies to Version 1, Release 7, Modification 0 of z/OS (5694-A01).

**© Copyright International Business Machines Corporation 2006. All rights reserved.**

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.





# Contents

<b>Preface</b> .....	xiii
How this text is organized .....	xiv
How each chapter is organized .....	xv
The team that wrote this redbook .....	xvi
Comments welcome .....	xvii
<b>Part 1. Introduction to networking on the mainframe</b> .....	1
<b>Chapter 1. Mainframes and networks</b> .....	3
1.1 Mainframes, networks, and you .....	4
1.2 Networks and online systems .....	5
1.3 Why are networks important? .....	7
1.4 Examples of mainframe-based networks .....	7
1.4.1 ATM cash withdrawal .....	8
1.4.2 Credit purchase at a retail store .....	9
1.4.3 Technology choices abound in network technology .....	9
1.5 Who supports the network? .....	10
1.6 What are the basic elements of a network? .....	11
1.7 Overview of mainframe network capabilities .....	12
1.7.1 z/OS Communications Server .....	15
1.7.2 SNA and TCP/IP on z/OS .....	17
1.8 Security in a network .....	17
1.8.1 Data protection .....	18
1.8.2 Availability .....	19
1.9 Summary .....	19
<b>Chapter 2. Network layers and protocols review</b> .....	21
2.1 Coming to terms .....	22
2.1.1 Layers .....	22
2.2 Physical media, layer 1 .....	23
2.2.1 Network interface card (NIC) .....	24
2.3 Data link layer, layer 2 .....	25
2.3.1 Ethernet .....	25
2.3.2 Media access control (MAC) addressing .....	27
2.4 Network layer, layer 3 .....	27
2.4.1 Address Resolution Protocol (ARP) .....	29
2.4.2 Network types .....	29
2.4.3 Local area network (LAN) .....	29
2.4.4 Wide area network (WAN) .....	30

2.4.5	Virtual LAN	30
2.4.6	Network routing	31
2.4.7	Building routing tables	34
2.4.8	Internet Control Message Protocol (ICMP)	36
2.4.9	Other layer 3 protocols	37
2.5	Transport layer, layer 4	37
2.5.1	Transmission Control Protocol (TCP)	37
2.5.2	User Datagram Protocol (UDP)	40
2.5.3	Sockets	40
2.6	Applications	41
2.7	Security	41
2.7.1	Firewalls and gateways	41
2.7.2	Security protocols	42
2.7.3	Protection	42
2.8	Systems Network Architecture (SNA)	42
2.9	Summary	43
<b>Chapter 3. Hardware connectivity on the mainframe.</b>		<b>45</b>
3.1	Connecting the mainframe	46
3.2	Channel subsystem (CSS)	46
3.2.1	Control units	48
3.2.2	Logical channel subsystem (LCSS)	48
3.3	The mainframe channel subsystem and network links	49
3.4	Hardware channels	51
3.5	Channel command word (CCW)	52
3.5.1	CCW-based operations	52
3.5.2	Parallel channel	52
3.5.3	Enterprise System Connectivity (ESCON) channel	53
3.5.4	Fiber connection (FICON)	55
3.5.5	Channel-attached network devices	55
3.6	Coupling channels	55
3.7	Open Systems Adapter (OSA)	56
3.7.1	Queued Direct I/O (QDIO)	58
3.7.2	Non-QDIO mode	61
3.7.3	OSC mode	61
3.7.4	Open Systems Adapter for NCP (OSN)	61
3.8	HiperSockets	63
3.9	Making the connections	65
3.10	Summary	67
<b>Chapter 4. Sample configuration</b>		<b>69</b>
4.1	Requirements for a mainframe network	70
4.2	The ZOS Company data center	70

4.2.1 Key mainframe network availability aspects . . . . .	73
4.3 Summary . . . . .	77
<b>Part 2. TCP/IP implementation on the mainframe . . . . .</b>	<b>79</b>
<b>Chapter 5. TCP/IP on z/OS . . . . .</b>	<b>81</b>
5.1 The heart of the matter . . . . .	82
5.1.1 Control issues and the stack . . . . .	82
5.1.2 IPv4 versus IPv6 . . . . .	83
5.2 The TCP/IP profile . . . . .	83
5.2.1 Profile statements . . . . .	84
5.3 The FTP server . . . . .	90
5.3.1 Character sets - FTP server . . . . .	93
5.4 The telnet daemon . . . . .	94
5.4.1 Character sets - telnet daemon . . . . .	95
5.5 A good resolver is hard to find . . . . .	95
5.5.1 Searching for resolver configuration information . . . . .	97
5.5.2 The multi-stack environment . . . . .	100
5.6 TCP/IP clients . . . . .	101
5.6.1 Character sets . . . . .	101
5.7 Summary . . . . .	102
<b>Chapter 6. TCP/IP in a sysplex . . . . .</b>	<b>103</b>
6.1 Computer cluster . . . . .	104
6.2 The z/OS sysplex . . . . .	104
6.2.1 Cross-system Coupling Facility (XCF) . . . . .	105
6.2.2 Workload Manager (WLM) . . . . .	105
6.3 Dynamic virtual addressing . . . . .	106
6.3.1 Unique application-instance DVIPA . . . . .	106
6.3.2 Multiple application-instance DVIPA . . . . .	110
6.3.3 Why DVIPA? . . . . .	112
6.4 Dynamic cross-system coupling . . . . .	112
6.4.1 VTAM and XCF . . . . .	112
6.4.2 Establishing the links . . . . .	113
6.4.3 More than just XCF . . . . .	114
6.5 Sysplex distributor . . . . .	115
6.5.1 What is a tightly-coupled host to do? . . . . .	116
6.5.2 Sysplex distributor in action . . . . .	117
6.5.3 Moving distribution to the network . . . . .	122
6.5.4 Problem detection and recovery in the cluster . . . . .	122
6.6 Routing in a sysplex . . . . .	123
6.6.1 Open Shortest Path First (OSPF) . . . . .	124
6.6.2 Network interface card . . . . .	126
6.7 Summary . . . . .	128

<b>Part 3. SNA and SNA/IP implementation on the mainframe</b> .....	131
<b>Chapter 7. Systems Network Architecture - basics and implementation</b>	133
7.1 What is Systems Network Architecture (SNA)? .....	134
7.1.1 History .....	134
7.1.2 What you need to know about SNA today .....	135
7.2 The evolution of SNA .....	136
7.3 Subarea networking .....	138
7.3.1 SNA nodes .....	138
7.3.2 System services control point (SSCP) .....	141
7.3.3 Subareas and domains .....	141
7.3.4 Connecting subarea nodes .....	143
7.4 Architectural components of the SNA network .....	146
7.5 VTAM subarea definitions .....	155
7.5.1 Start options .....	155
7.5.2 Configuration lists .....	156
7.5.3 Defining resources to VTAM .....	157
7.5.4 Sample network .....	159
7.6 Summary .....	162
<b>Chapter 8. SNA Advanced Peer-to-Peer Networking (APPN)</b> .....	165
8.1 Introduction .....	165
8.1.1 Advanced Program-to-Program Communications (APPC) .....	167
8.1.2 Advanced Peer-to-Peer Networking (APPN) .....	167
8.1.3 APPC versus APPN .....	168
8.2 APPN node types .....	169
8.2.1 Low-entry networking (LEN) nodes .....	169
8.2.2 End nodes (EN) .....	170
8.2.3 Network nodes (NN) .....	171
8.2.4 Specialized network node types .....	173
8.2.5 Specialized VTAM Nodes .....	176
8.3 Control point (CP-CP) sessions .....	179
8.3.1 End node CP-CP sessions .....	180
8.3.2 Network node CP-CP sessions .....	180
8.3.3 NETID considerations .....	181
8.4 APPN databases .....	181
8.4.1 Topology database (TOPO DB) .....	182
8.4.2 Directory services database (DS DB) .....	183
8.5 APPN network topology .....	184
8.5.1 Topology data update (TDU) flows .....	184
8.5.2 Locating resources .....	187
8.5.3 Route calculation .....	193
8.5.4 Transmission group (TG) characteristics .....	194

8.5.5 Class of service (CoS) . . . . .	194
8.5.6 Choosing session paths . . . . .	195
8.6 High performance routing (HPR) . . . . .	196
8.6.1 Rapid transport protocol (RTP) . . . . .	197
8.6.2 Automatic network routing (ANR) . . . . .	198
8.6.3 HPR headers . . . . .	200
8.6.4 Path switching . . . . .	201
8.7 Connection networks . . . . .	202
8.8 Dependent LU requester/server (DLUR/DLUS) . . . . .	204
8.9 Defining a VTAM APPN network . . . . .	206
8.10 Summary . . . . .	209
<b>Chapter 9. SNA/IP implementation . . . . .</b>	<b>211</b>
9.1 Background . . . . .	212
9.1.1 Why preserve SNA applications? . . . . .	212
9.1.2 SNA applications and integration methods . . . . .	213
9.2 DLSw and Logical Link Control 2 (LLC2) . . . . .	214
9.2.1 Establishing an LLC2 connection over a LAN . . . . .	214
9.2.2 Data link switching (DLSw) . . . . .	215
9.2.3 DLSw over the WAN . . . . .	215
9.2.4 Connecting SDLC devices using DLSw . . . . .	217
9.2.5 DLSw configuration . . . . .	220
9.3 Enterprise Extender . . . . .	220
9.3.1 What is it? . . . . .	220
9.3.2 Why does Enterprise Extender use UDP packets? . . . . .	225
9.3.3 Class of service (CoS) . . . . .	227
9.3.4 EE implementation in non-z/OS remote sites . . . . .	228
9.3.5 Internet connectivity exploitation . . . . .	228
9.4 Comparing Enterprise Extender and DLSw . . . . .	229
9.5 Implementing Enterprise Extender . . . . .	230
9.6 Configuration examples . . . . .	233
9.6.1 Enterprise Extender configuration . . . . .	234
9.6.2 Extended border node configuration . . . . .	237
9.6.3 Cisco SNASw definitions . . . . .	239
9.7 Summary . . . . .	239
<b>Chapter 10. TN3270 Enhanced . . . . .</b>	<b>241</b>
10.1 Introduction . . . . .	242
10.1.1 3270 data stream . . . . .	243
10.2 TN3270 Enhanced (TN3270E) . . . . .	246
10.2.1 Telnet protocol and SNA meet . . . . .	246
10.2.2 TN3270E description . . . . .	247
10.2.3 Additional functions supported in z/OS . . . . .	249

10.3	Positioning a TN3270E server . . . . .	250
10.4	Implementing the TN3270E server . . . . .	250
10.4.1	TELNETPARMS statement block . . . . .	251
10.4.2	BEGINVTAM statement block . . . . .	253
10.4.3	VTAM setup for TN3270E server . . . . .	256
10.5	Summary . . . . .	257
<b>Part 4. Network operations and administration . . . . .</b>		<b>259</b>
<b>Chapter 11. Operational procedures . . . . .</b>		<b>261</b>
11.1	Operational tasks . . . . .	262
11.1.1	Network startup . . . . .	263
11.1.2	Stopping the network . . . . .	267
11.2	z/OS network administrator tasks . . . . .	268
11.2.1	Managing VTAM . . . . .	269
11.2.2	Managing TCP/IP . . . . .	273
11.2.3	Controlling TCP/IP applications . . . . .	277
11.3	Documenting the network environment . . . . .	280
11.4	Summary . . . . .	282
<b>Chapter 12. Network security . . . . .</b>		<b>283</b>
12.1	Security . . . . .	284
12.1.1	The context of security . . . . .	284
12.2	Elements of security . . . . .	285
12.3	TCP/IP security . . . . .	286
12.3.1	Industry standard security features . . . . .	287
12.3.2	TCP/IP on z/OS security features . . . . .	294
12.4	TN3270 security . . . . .	299
12.5	SNA security . . . . .	300
12.5.1	Subarea . . . . .	300
12.5.2	APPN security . . . . .	301
12.6	Summary . . . . .	301
<b>Chapter 13. Problem determination . . . . .</b>		<b>303</b>
13.1	Network problems . . . . .	304
13.1.1	Determining the problem . . . . .	304
13.2	Tools and diagnostic aids . . . . .	305
13.2.1	Common (z/OS-wide) tools and diagnostic aids . . . . .	305
13.2.2	VTAM tools and diagnostic aids . . . . .	308
13.2.3	TCP/IP tools and diagnostic aids . . . . .	313
13.3	VTAM problem determination . . . . .	318
13.4	TCP/IP problem determination . . . . .	320
13.4.1	Messages . . . . .	321
13.4.2	TCP/IP server or client address space problems . . . . .	321

13.4.3 Diagnosing network problems .....	322
13.5 Communications Storage Manager (CSM) .....	324
13.5.1 Debugging CSM .....	324
13.6 Performance and tuning .....	326
13.7 Summary .....	327
<b>Notices</b> .....	329
Trademarks .....	330
<b>Glossary</b> .....	331
<b>Index</b> .....	359



# Preface

This IBM® Redbook is designed for a one-semester course intended to provide information systems students with the background knowledge and skills necessary to begin using the basic communications facilities of a mainframe-based system. It provides a broad understanding of networking principles and the hardware and software components necessary to allow the mainframe to participate in a high volume data communications network. It is part of a planned series of textbooks designed to introduce students to mainframe concepts and help prepare them for a career in large systems computing.

While many of the networking concepts covered are operating system-independent, the main emphasis is on the z/OS® operating system. You are assumed to have successfully completed introductory courses in computer system concepts, including courses in computer organization and architecture, operating systems, data management and data communications, and systems design and analysis.

A basic understanding of z/OS job control, library structure, and system libraries is assumed. It is strongly recommended that you have already completed an introductory course on z/OS, such as one that uses *Introduction to the New Mainframe: z/OS Basics* or a comparable text. In some instances, the *z/OS Basics* course and the course associated with this textbook could be taken together.

This book can also be used as a prerequisite for courses in advanced topics, or for internships and special studies. However, it does not comprehensively cover all aspects of data communications, nor is it a reference book that discusses every feature and option of the zSeries communications facilities.

Others who will benefit from this text include data processing professionals who have experience on non-mainframe-based platforms, or who are familiar with some aspects of the mainframe environment or applications, but want to learn about the security and integrity facilities and benefits of the mainframe environment.

As we go through this book, we suggest that the instructor alternate between text, lecture, discussions, and hands-on exercises. Many of the exercises are cumulative, and are designed to show the student how to design and implement the topic presented. The instructor-led discussions and hands-on exercises are an integral part of the course material, and can include topics not covered in this

textbook. We use simplified examples and focus mainly on basic system functions. Hands-on exercises are provided throughout the course to help students explore the mainframe style of computing.

At the end of this course, you will have learned:

- ▶ Concepts of mainframe-based data communications
- ▶ Methods for implementing TCP/IP on the z/OS operating system
- ▶ Methods for implementing SNA on the z/OS operating system
- ▶ Basic skills in network operations, security, and problem determination

## How this text is organized

This text is organized in four parts, as follows:

Part 1, “Introduction to networking on the mainframe” on page 1, provides an overview of the importance of the mainframe environment and some of the functions and roles of networking professionals. It also introduces you to several typical applications and environments that are used throughout the text as examples when discussing specific networking scenarios.

- ▶ Chapter 1, “Mainframes and networks” on page 3, introduces mainframe networking, the latest networking technologies, and the role of the networking administrator.
- ▶ Chapter 2, “Network layers and protocols review” on page 21, reviews network layers, the protocols at each layer, and the hardware facilities that support data transfer.
- ▶ Chapter 3, “Hardware connectivity on the mainframe” on page 45, describes the various ways a mainframe connects to a network.
- ▶ Chapter 4, “Sample configuration” on page 69, describes a sample configuration that is used to demonstrate various course concepts.

Part 2, “TCP/IP implementation on the mainframe” on page 79, examines the use of IP-based communications in a network-attached z/OS system and among multiple systems in a z/OS cluster or Parallel Sysplex®.

- ▶ Chapter 5, “TCP/IP on z/OS” on page 81, explains the way TCP/IP is implemented on z/OS.
- ▶ Chapter 6, “TCP/IP in a sysplex” on page 103, explains the special features of TCP/IP in a Parallel Sysplex.

Part 3, “SNA and SNA/IP implementation on the mainframe” on page 131, examines IBM’s System Network Architecture, including its evolution and its current integration into IP networks.

- ▶ Chapter 7, “Systems Network Architecture - basics and implementation” on page 133, describes the first of two SNA implementations, which is subarea networking.
- ▶ Chapter 8, “SNA Advanced Peer-to-Peer Networking (APPN)” on page 165, describes the second SNA implementation, which is Advanced Peer-to-Peer Networking® (APPN).
- ▶ Chapter 9, “SNA/IP implementation” on page 211, explains technologies and solutions used to integrate SNA and IP networks.
- ▶ Chapter 10, “TN3270 Enhanced” on page 241, shows how the mainframe display management protocol (the 3270 data stream) is integrated into IP networks.

Part 4, “Network operations and administration” on page 259, examines network operations, security, and problem determination.

- ▶ Chapter 11, “Operational procedures” on page 261, provides an overview of operational tasks that a z/OS network administrator performs. The chapter includes network commands and displays commonly used to monitor and control the network.
- ▶ Chapter 12, “Network security” on page 283, explains how mainframe networking implements security.
- ▶ Chapter 13, “Problem determination” on page 303, provides information about diagnosing problems in the mainframe network.

## How each chapter is organized

Each chapter follows a similar format:

- ▶ Objectives for the student
- ▶ Topics that teach a central theme related to mainframe networking
- ▶ Summary of the main ideas of the chapter
- ▶ A list of key terms introduced in the chapter

## 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.

**Matt Nuttall** is an information technology specialist with IBM Global Services in Canada. He has over 16 years of experience in information technology, most of it in networking. Matt holds a degree in Computer Science from the University of British Columbia. His areas of expertise include networking and networking security. He has written several IBM Redbooks™ relating to z/OS.

**Micky Reichenberg** is an independent consultant with more than 30 years of experience in mainframe networking. He specializes in mainframe TCP/IP, SNA, and open systems connectivity to the mainframe. Prior to becoming a consultant, Micky was a systems programmer with IBM in Israel for 17 years. During his assignment with the International Technical Support Center (ITSO) in Raleigh, North Carolina, he published five networking-related Redbooks. He holds a Bachelor's degree in Aeronautical Engineering from the Technion - Israel Institute of Technology.

**Chris Hastings** is a Senior Software Engineer at IBM Poughkeepsie. He has 25 years of experience developing user assistance manuals and on-line help for large operating systems, such as z/VM® and z/OS. Chris holds a Master of Arts degree in Philosophy from Duquesne University, Pittsburgh, Pennsylvania.

Additional contributors were:

**Grant Cuming** is a Senior Systems Programmer with IBM Global Services in New Zealand. He has more than 20 years of experience working with government, private companies, and IBM outsourced accounts. Grant specializes in implementing and maintaining z/OS TCP/IP, VTAM®, and networking-related components and solutions.

**Joel Porterie** is a Senior IT Specialist who has been with IBM France for 28 years. He works for Network and Channel Connectivity Services in the EMEA Product Support Group. His areas of expertise include z/OS, TCP/IP, VTAM, OSA-Express and Parallel Sysplex for zSeries. Joel has taught OSA-Express and FICON® problem determination classes and provided on site assistance in these areas in numerous countries. He also co-authored the IBM Redbooks *Using the IBM S/390 Application StarterPak*, SG24-2095, *OSA-Express Gigabit Ethernet Implementation Guide*, SG24-5443, and *OSA-Express Implementation Guide*, SG24-5948.

**Mike Ebbers** has worked with mainframe systems at IBM for 32 years. For part of that time, he taught hands-on mainframe classes to new IT professionals who

were just out of college. Mike currently creates IBM Redbooks, a popular set of product documents that can be found at:

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

Thanks to the following for contributing to this project:

**Terry Barthel**, editor  
International Technical Support Organization, Poughkeepsie Center

## Comments welcome

Your comments are important to us!

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

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

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

- ▶ Send your comments in an e-mail to:

[redbook@us.ibm.com](mailto:redbook@us.ibm.com)

- ▶ Mail your comments to:

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





# Part 1

# Introduction to networking on the mainframe





# Mainframes and networks

**Objective:** As a data communications expert in the world of mainframe computing, you will need to understand the role of the network in your company's business objectives and corporate infrastructure. You also need to understand how the latest networking technologies work with your company's mainframe computer.

After completing this chapter, you will be able to:

- ▶ Explain the use of networks in high volume industry transaction processing.
- ▶ List at least three typical advantages of a mainframe in network communications.
- ▶ Describe the role of a network administrator in a large network.
- ▶ List the major software components of the z/OS Communications Server.

## 1.1 Mainframes, networks, and you

In the broadest sense of the word, a *network* is an interconnected system of people or things. In the fast-paced, lively field of information technology (IT), a network is defined as the hardware and software that enables computers to share files and resources and exchange data. Depending on the size of a business, a network can be as simple as two personal computers on a locally connected network or as complex as the Internet, a worldwide network of millions of computers of various types.

To send or receive data through a network, a company's employees interact through a variety of communication devices, such as telephones, workstations, and computers. Network data can flow through an even greater variety of mechanisms: communication software and hardware, telephone wires, broadband cable, wireless and microwave transmission units, satellite, fiber optics, and so on.

Regardless of which elements comprise a particular network, you—the end user—are the ultimate source and destination of the information that flows through it.

To some extent, the definition of “network” depends upon *who* is using the network. For example, even though voice and data share the same network, an IT professional hired to support the voice traffic will likely view the network differently than the person assigned to maintain data traffic. The telephony expert or electrical engineer might describe the network as “a group or system of electronic components and connecting circuitry designed to function in a specific manner,” while a network designer or architect might explain the network as “a system of lines or channels that cross or interconnect, forming a complex, interconnected group or system.”

In this text, we'll try not to be so tedious or evasive. Our definition of “network” encompasses all the usual ideas:

- ▶ A group of interconnected computers capable of exchanging information
- ▶ A collection of computers and associated devices connected by communications facilities (hardware and software) that share information
- ▶ The entity that allows users, applications, and computers in a corporation to exchange data and files for the purpose of transacting business

And, our primary focus will be on how network technology relates to mainframe computers, the workhorses of corporate IT.

**What is a mainframe?** It's a computer that supports dozens of applications and input/output devices to serve tens of thousands of users simultaneously.

What separates the mainframe from other computers is not just its processing capabilities. A mainframe has redundant features and system health awareness capabilities that enable it to deliver 99.999% availability.

Throughout this text, the general term “mainframe” refers to large computers like those in the IBM System z9™ and eServer™ zSeries processor families.

To be effective, corporate communications rely on secure links between thousands of end users, applications, and computers of various sizes, including mainframes. Wherever speed and security are essential, mainframes are used as the most critical servers in the network infrastructure.

If you have never pondered the incredible inter-connectedness of the modern world, its computers, and its end users, consider your own experience: you use a complex network when you:

- ▶ Withdraw money from a bank account through an automated teller machine (ATM)
- ▶ Submit a payment at the supermarket with a debit or credit card
- ▶ Purchase a music download over the Internet

Computer networks touch nearly every aspect of everyday life. And, when a large organization needs transaction processing, the odds are that the network is connected to a mainframe.

## 1.2 Networks and online systems

Networks are categorized as internets, intranets and extranets:

- ▶ *Internet* is a collection of individually managed networks, connected by intermediate networking devices, that function as a single large network. *Internetworking* refers to the industry, products, and procedures that help to create and administer internets.
- ▶ *Intranet* is a privately maintained computer network that can be accessed only by authorized persons and is limited to one institution.
- ▶ *Extranet* is an extension of an institution's intranet, used to connect business partners. In today's IT environment, the World Wide Web is the enabler for communication between the institution, business partners, and people it deals with, often by providing limited access to its intranet.

Before the advent of the Internet, employees in a corporation perceived the network as the *terminals* that served the company's business transactions. This workload was rather predictable both in transaction rate and mix of transactions, and much of the work could be done after hours through batch processing on the mainframe.

The paradigm used today is online transaction processing (OLTP). OLTP is a class of program that facilitates and manages transaction-oriented applications, typically for data entry, order entry and retrieval transactions in a number of industries, including banking, airlines, mail order, supermarkets, and manufacturers. Probably the most widely installed OLTP product, excluding the Web servers that front-end most OLTP, is the IBM Customer Information Control System, or CICS® (pronounced “kicks”).

Today's online transaction processing increasingly requires support for transactions that span a network and may include more than one company. For this reason, new OLTP software uses client/server processing and brokering software that allows transactions to run on different computer platforms in a network.

Today's networks and transactional systems must be able to support an unpredictable number of concurrent users and transaction types. Most transaction programs need to respond in short time periods—fractions of a second in some cases.

For example, inside a bank branch office or through the Internet, customers are using online services when checking an account balance or transferring fund balances.

In fact, an online transaction system has many of the characteristics of an operating system:

- ▶ Managing and dispatching tasks
- ▶ Controlling user access authority to system resources
- ▶ Managing the use of real memory
- ▶ Managing and controlling simultaneous access to data files
- ▶ Providing device independence

Most of the traffic in a network involves transaction processing where one side initiates the transaction and the other side processes, authorizes, and approves or declines the transaction.

Examples of activities that result in network traffic include:

- ▶ Ordering and receiving parts to assemble automobiles

- ▶ Cash withdrawal from an automated teller machine (ATM)
- ▶ Purchasing merchandise at a retail point-of-sale (POS)
- ▶ Paying bills over the Web using a home banking application
- ▶ Receiving loan approval to buy a home
- ▶ Such e-business as flight and car rental reservations

In fact, even receiving a traffic citation can generate network traffic. How else can the patrol officer check for outstanding warrants?

## 1.3 Why are networks important?

In today's competitive market, responsiveness to customer or supplier demand is often a decisive factor in the success of an organization. The network is considered one of the most critical resources in an organization, both in the private and public sectors.

Networks are created to provide a means to satisfy an objective or need. These objectives and needs are frequently critical, therefore the network itself is critical. Consider the metaphor of a transportation network (roads, highways, rails, and so on). If any of these conduits were to become suddenly unavailable, our ability to distribute food, clothes and products would be seriously compromised. The residents of a town or country who need the food, clothes and products are the "end users" of this particular type of network.

Similarly, a computer network is created to provide a means of transmitting data, sometimes essential data, from one computer to another. The accuracy and speed of daily business transactions for large organizations are vital to their success. Unscheduled disruption resulting in the failure to process these daily business transactions are costly and potentially disastrous.

The widespread use of networks extends the reach of organizations. These remote interactions with customers, suppliers and business partners have significantly benefited countless businesses. It has correspondingly positively impacted the overall productivity of many countries. Such productivity gains, however, are only as good as the network.

## 1.4 Examples of mainframe-based networks

Mainframes are used by large organizations as their central transaction processing system. Transaction processing in this context requires high availability, security, performance, and responsiveness. For example, consumers

expect to be able to use their credit card 24 hours a day, 365 days a year. They expect those transactions to be safe and they don't expect to be left standing at the checkout waiting for it to all happen. The mainframe is specifically designed to be the "best of breed" for performing massive concurrent transaction processing in the range of hundreds of transaction per second.

In the examples that follow, we look at typical cases of networks as they are commonly used in high volume business transactions. Each of these examples shows an industry that relies on messages being sent electronically over a communication network. In most cases, a mainframe is used to send the message, one or more mainframes may be needed to route it to the appropriate place, and a third mainframe is used to receive it.

Although simplified to some extent, these examples provide some insight into the extent and complexity of electronic communication networks:

- ▶ ATM cash withdrawal
- ▶ Credit purchase at a retail store

In practice, the number of transactions, the interfaces among the business partners, and the number of data elements is several orders of magnitude more complex.

### **1.4.1 ATM cash withdrawal**

The simple act of withdrawing cash from an automated teller machine (ATM) is much more complicated than it appears. You begin by inserting your identification card and entering a personal identification number (PIN). Your identity is verified online when a computer in the network compares the information you entered to a database of customers belonging to that financial institution. Internal electronic messages are created to access the specific checking or savings account where the money is held. Then, the account balance is verified and approved. Finally, a message is sent back to the ATM to disperse the funds or refuse the transaction.

The withdrawal transaction triggers secondary transactions to update the appropriate checking or savings accounts; this is usually done in real-time. By the time the money is dispensed from the machine, the account balance will reflect the withdrawal. It becomes more complex if you make an out-of-territory withdrawal. For example, you use Bank 1's ATM to withdraw money from your account at Bank 2. The peer bank's database must be accessed and the account status verified.

All of this occurs as the customer waits at the machine. The network and mainframe computers involved must be very fast to keep the response time "reasonable" from the customer's point of view.

The successful completion of the transaction depends on, among other things, both banks using compatible network technology to exchange information.

### **1.4.2 Credit purchase at a retail store**

When you use a credit card to purchase goods from a retailer, then a network, and most likely a mainframe computer, is involved. When your credit card is electronically scanned, the identification is initially handled by the company (Bank) that provides the point of sale credit card reader. From there, the transaction is sent through the network to the credit card company's mainframe. When your account is validated and the transaction is approved, the credit card company issues a debit message to the issuing bank. Concurrently, a credit message to the merchant is issued.

The advantage of sending the transaction immediately is to detect whether you are exceeding your credit limit, and to prevent such violations. Furthermore, if the card is stolen, or if you have exceeded the credit limits, the merchant must be notified in time to void the purchase. Often, an intermediate host is used to handle and approve or disapprove the transaction. All of this can only be effective when a robust, responsive communication network is in place among the merchant, credit card company, and the issuing bank.

The transactions that were described take advantage of the following functionality that the mainframe can provide to an OLTP system:

- ▶ Availability - Customers do not expect an ATM to be unavailable. Ever.
- ▶ Security - The PIN number entered is encrypted at the ATM and decrypted at the host that executed the ATM transaction.
- ▶ Responsiveness - How long is a customer willing to wait until the transaction is completed?

### **1.4.3 Technology choices abound in network technology**

It is unlikely that each and every business supplier in these examples uses the same network components. Many may use IBM's Systems Network Architecture (SNA) protocols while most will use the TCP/IP protocols. An even smaller number may use proprietary protocols (protocols not standard in the industry).

When two or more partners do not use identical network components, there must be some process to enable them to coexist and to interpret (translate) each others' messages. In this course, the technology options available to design and implement a network to handle business transactions will be explored and applied to the above examples of business transactions.

Some of these products exist primarily to allow different protocols to function together. In particular, SNA is rapidly adapting to the IP-centric networks favored by today's organizations.

## 1.5 Who supports the network?

Network communications has both a software and a hardware aspect, and a separation of software and hardware administrative duties is common in large enterprises. The network administrator, a skilled software data communication expert, however, needs to understand both aspects.

The network administrator must bring a thorough understanding of the operating system's communications interfaces to any project that involves working with the company's network. While network hardware technicians have specific skills and tools for supporting the physical network, their expertise often does not extend to the operating system's communications interfaces. When a nationwide retail chain opens a new store, the network administrators and network hardware technicians must coordinate their efforts to open the new store.

Most of this textbook focusses on z/OS network concepts, implementations, and hardware. You may also find it useful to have a working knowledge of other operating systems available for mainframes.

One reason for this is that a given mainframe computer might run multiple operating systems. For example, the use of z/OS, z/VM, and Linux® on the same mainframe is common.

The responsibilities of a z/OS network administrator may include:

- ▶ Defining, maintaining, and modifying an existing mainframe network
- ▶ Providing technical guidance to application development and business unit projects
- ▶ Determining, isolating, and correcting problems
- ▶ Tuning performance
- ▶ Making planning recommendations for capacity
- ▶ Developing operational procedures
- ▶ Training network operators
- ▶ Maintaining an awareness of emerging network technologies
- ▶ Recommending and implementing new network technologies

This course is intended to assist in preparing you to fulfill these responsibilities as a member of the z/OS networking team in a large enterprise.

## 1.6 What are the basic elements of a network?

Basic elements of a computer network include hardware, software, and protocols. The interrelationship of these basic elements constitutes the infrastructure of the network. To revisit the transportation metaphor we used earlier, if we think of a network as roads, highways, rails, and other means of transport, the network protocols are the “traffic rules.”

A network infrastructure is the topology in which the nodes of a local area network (LAN) or a wide area network (WAN) are connected to each other. These connections involve equipment like routers, switches, bridges and hubs using cables (copper, fiber, and so on) or wireless technologies (Wi-Fi).

The network protocols define how two devices in the network communicate. The specification of the network protocols starts with the electrical specifications of how a networking device is connected to the infrastructure. For example, line voltage levels, carrier signals and the designation of which line might be used for what types of signals must all be specified. Building up from there, network protocols include such specifications as the methods that can be used to control congestion in the network and how application programs will communicate and exchange data.

A popular method of documenting network protocols is to use a *layered network architecture* model. Network architecture models separate specific functions into *layers*, which collectively form a *network stack*. While a protocol consists of rules that define characteristics for transporting data between network nodes, the layered model separates the end-to-end communication into specific functions performed within each layer.

Ideally, the layers are isolated from each other—each layer does not need to know how the layer below it functions. All a layer needs to know is how to interact with the layers adjacent to it. You can learn more about network layers in Chapter 2, “Network layers and protocols review” on page 21.

Today, TCP/IP is by far the most dominant suite of networking protocols. Prior to TCP/IP, SNA was arguably the dominant protocol suite. There is some irony here, because TCP/IP is the older of the two protocols. Many networks in larger organizations are using both of these protocol suites. As with most networking protocols, both SNA and TCP/IP are layered protocol stacks.

## 1.7 Overview of mainframe network capabilities

IBM's current mainframe technology provides significantly large servers with a distinctive strength of handling a high volume of transactions and input/output operations in parallel. The mainframe is capable of serving a large number of network nodes geographically dispersed across the world while simultaneously handling a high volume of input and output operations to disk storage, printers, and other attached computers.

Mainframe architecture includes a variety of network capabilities. Some of these capabilities include:

- ▶ IP communication among large numbers of Linux and z/OS operating systems running as z/VM (Virtual Machine) guest machines

**Note:** What is a z/VM guest machine? z/VM is another mainframe operating system that, on its own, does nothing more than reproduce the instruction set of a mainframe machine. It provides a guest operating system with a self-contained environment that appears to the guest as though it were a real physical machine. z/VM requires very low overhead to produce guest machines and can consequently support very large numbers of them (tens of thousands).

- ▶ IP communication among independent operating systems running in logical partitions (LPARs) on the same machine
- ▶ IP communications among a tightly coupled cluster of mainframe LPARs (called a Parallel Sysplex)
- ▶ Communications via the TCP/IP suite of protocols, applications, and equipment (for example, the Internet, intranets, and extranets)
- ▶ System Network Architecture (SNA) suite of protocols and equipment, including subarea and Advanced Peer-to-Peer Networking with high performance routing (APPN/HPR)
- ▶ Integration of SNA into IP networks using Enterprise Extender (EE) technology

If you are unfamiliar with some of these terms, this is to be expected. Subsequent chapters will discuss these protocols and much more.

Figure 1-1 on page 14 illustrates a typical but simplified mainframe-based network. The following text refers to this figure.

The mainframe is usually connected to the outside world using an integrated LAN adapter called the Open Systems Adapter-Express (OSA-Express). The

OSA-Express is the equivalent of the network interface card used in Windows® and UNIX® systems. It supports various operational modes and protocols. Most commonly, the OSA-Express card will use the Ethernet protocol, running over copper wire or fiber optic cabling. The latest OSA-Express card, called OSA-Express2, supports Ethernet at a speed of 10 Gb/s.

Because the I/O subsystem of the mainframe is different from Intel® or UNIX systems, the OSA card implements advanced technologies required for networking.

The OSA-Express card is connected to a redundant backbone switch/router (either in a server farm or dedicated to the mainframe) that implements the connection to the outside world (as shown in Figure 1-1 on page 14).

**Note:** A redundant backbone switch or router is used to connect critical business servers to the primary (or most important) network for a given organization. The switch or router provides redundancy by providing more than one physical path to the backbone network. The switch or router also is aware of the network through a routing protocol, which ensures that changes to the network are quickly and seamlessly accommodated.

The backbone network itself is an organization's high-traffic density network.

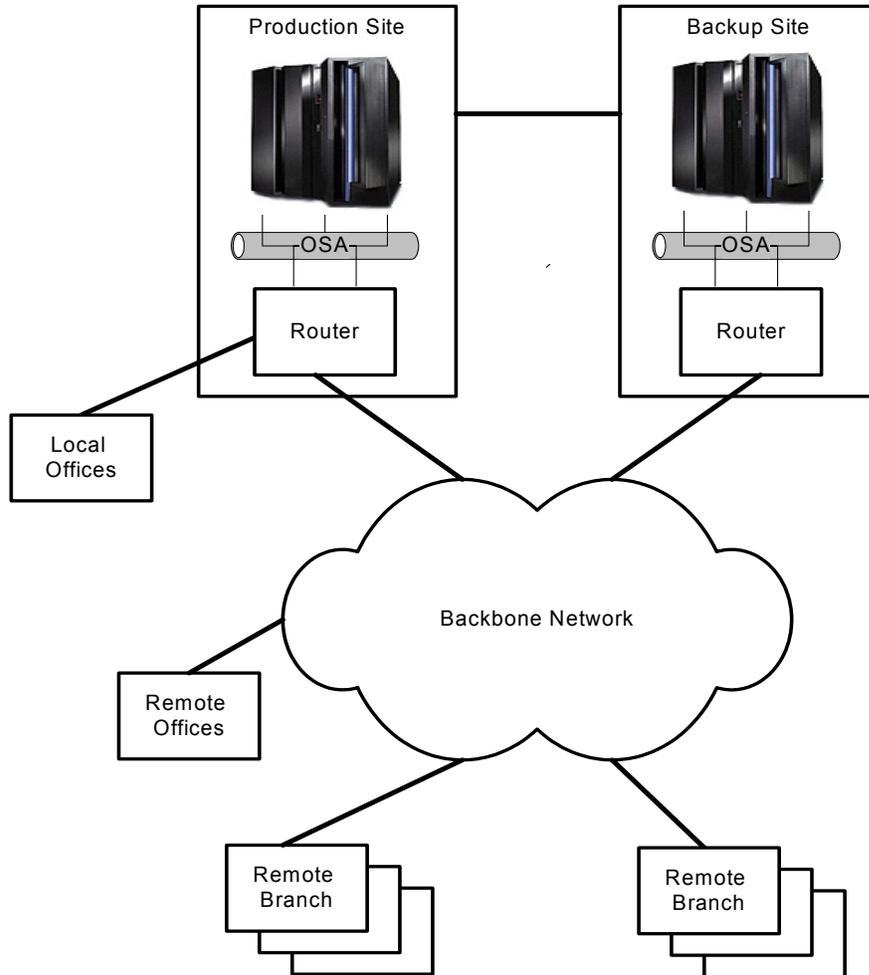


Figure 1-1 Typical mainframe network

A backup site takes over the data processing for planned and unplanned outages of the production site. The backup site is self-contained and can provide data processing services for a long time. Duplicating the production site can be very costly. The level and the type of services the backup site will provide is determined by the cost of a backup compared to the cost of a failure. The larger the organization, the higher the cost of failure and hence the greater value placed upon a fully functional backup site.

The backup and the production site are connected using high speed connections, normally using fiber optics. In addition to networking related data,

the connections are used to mirror data stored on disks from the production site at the backup site. Mirroring may be done in real-time.

Offices used for the computer personnel, administration, and back office services are usually located in the vicinity of the production computer site. These locations may be in the same building, the same campus, or a few blocks away. These sites also would be connected using high speed connections.

Remote sites, such as branch offices and remote offices, are connected to the backbone network. The backbone network might use carrier-supplied communication lines. The speed, the protocol, and the topology are designed and implemented by the networking department and the network users.

**Note:** A carrier-supplied network is a network that is provided (maintained, supported and operated) on behalf of another organization. It is a form of outsourcing: an organization simply needs the network, so it enlists another organization to supply the network.

### 1.7.1 z/OS Communications Server

The z/OS operating system includes a software component called z/OS Communications Server. z/OS Communications Server implements the SNA and TCP/IP protocols. SNA applications and transaction servers (like CICS) can use SNA or TCP/IP to send and receive data. Industry standard internet applications can use TCP/IP to send and receive data. For example, a z/OS server may run FTP, telnet, Web servers (HTTP), and mail programs (Simple Mail Transfer protocol, or SMTP).

z/OS Communications Server provides a set of communications protocols that support connectivity functions for both local and wide-area networks, including the Internet. z/OS Communications Server also provides performance enhancements that can benefit a variety of well-known TCP/IP applications. These performance enhancements, which may be software-based or hardware-based, are discussed in their appropriate contexts later in this book.

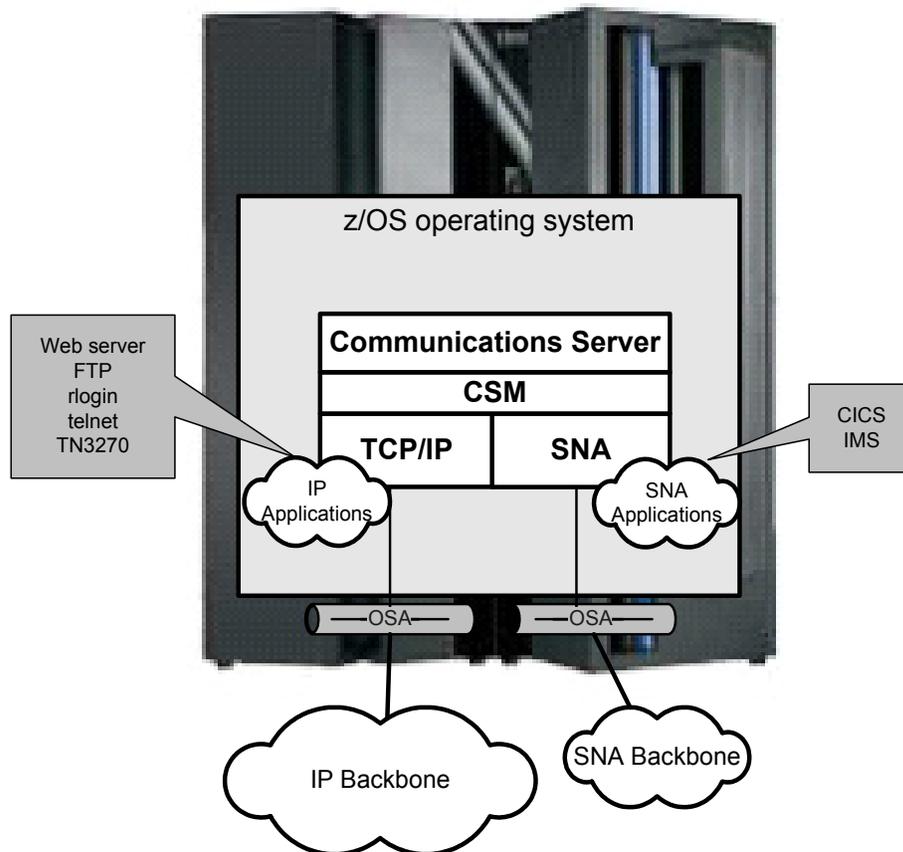


Figure 1-2 z/OS Communications Server

As shown in Figure 1-2, z/OS Communications Server includes three major components, which are:

- ▶ The TCP/IP protocol stack.
- ▶ The SNA protocol stack contained in Virtual Telecommunications Access Method (VTAM).
- ▶ The Communications Storage Manager (CSM), which provides a shared I/O buffer area for both TCP/IP and VTAM data flow. The CSM function allows authorized host applications to share data without having to physically move the data.

Similar to TCP/IP functions, SNA functions are implemented on a number of platforms besides z/OS, for example, AIX®, AS/400®, Microsoft® Windows, and Linux. As a result, z/OS application programmers can exploit technological advancements in communications (information access, electronic commerce, and collaboration) across distinctly different operating systems.

### 1.7.2 SNA and TCP/IP on z/OS

In the past, a mainframe backbone network used SNA. With the prevalence of TCP/IP and the introduction of SNA/IP integration technology and additional tools, current mainframe networks are migrating to IP-based networks.

SNA was developed by IBM for the business community. SNA provided industry with a technology that permitted unparalleled business opportunities. What TCP/IP and the Internet were to the public in the 1990s, SNA was to large enterprises in the 1980s. TCP/IP was widely embraced when the Internet came of age because it permitted access to remote data and processing for a relatively small cost. TCP/IP and the Internet resulted in a proliferation of small computers and communications equipment for chat, e-mail, conducting business, and downloading and uploading data.

Large SNA enterprises have recognized the increased business potential of expanding the reach of SNA-hosted data and applications to this proliferation of small computers and communications equipment in customers' homes and small offices.

**Note:** So, why isn't the Internet running SNA protocols? What happened? The answer is simple: complexity. SNA is a deterministic architecture. It uses a hierarchical method of definitions and leaves very little to chance. Bandwidth, connections, and users all need to be predefined completely, or at least to some degree. Contrast this to IP, in which nothing is predetermined and a large amount of unpredictability exists within bandwidth, connectivity, and usage.

## 1.8 Security in a network

Today more than ever, businesses depend on the critical data that flows over networks. A large amount of sensitive and confidential data is stored and retrieved from z/OS systems, so the data that moves through z/OS-attached networks must be secured, have high integrity, and be available at all times. The mainframe environment includes both hardware and software tools that meet these goals.

New mainframe hardware and software are ideal for network transactions because they are designed to allow huge numbers of users and applications to access rapidly and simultaneously the same data without interfering with each other. In networks that support thousands of end users, the mainframe concepts of data integrity, security and reliability are extended to include the network.

The designer of a large network must balance the need for data and transaction security with the requirement to provide rapid response time and reliability and availability of the network.

Some of the aspects of security that need to be taken into account are discussed briefly below. This topic is discussed in more detail in Chapter 12, “Network security” on page 283.

### 1.8.1 Data protection

Data protection not only includes privacy, but also *integrity*. For example, a financial transaction should be kept confidential no matter where it exists on a network. But, just as importantly, there must be controls in place to ensure that the data has not been altered.

A side issue of data protection is non-repudiation: there must be a mechanism in place to ensure that a sender cannot deny having sent a packet. Conversely, non-repudiation requires a mechanism such that a receiver cannot deny having received a packet (a *packet* is a string of data characters). Again, it is paramount for a financial institution to be able to confirm that a transaction has genuinely been sent by who we believe sent it, and that it has been received by who we expect to receive it.

The networking protocols such as TCP have built-in services which guarantee that data sent from an application arrives at its destination in the same sequence as it was transmitted and is error-free. By error-free, we mean that the same bit sequence that was transmitted is delivered to the destination node. The lower two layers in the networking architecture have the responsibility for the bit sequence and the transport layer has the responsibility for the correct sequence.

To implement these network design goals, z/OS and affiliated products provide these services:

- ▶ z/OS system and resource security is provided by both the IBM Security Server and the z/OS Communications Server components. IBM Security Server includes Resource Access Control Facility (RACF®) for authentication, authorization, and restriction.
- ▶ The z/OS Communications Server components (VTAM and TCP/IP) each include parameters to encrypt network traffic. For example, TCP/IP includes

firewall filtering, Virtual Private Network (VPN), and Transport Layer Security (TLS) capabilities as part of the protocol stack itself.

- ▶ Each of the major IBM subsystems used for deploying business applications, such as Customer Information Control System (CICS), Database 2™ (DB2®), Information Management System (IMS™), WebSphere® Application Server, HTTP Server, Message Queuing Series (MQSeries®), and so forth, in conjunction with RACF and other mainframe components, have security mechanisms available that provide additional levels of security.

Each of the available tools for securing resources and data can be used independently or together to accomplish security objectives.

## 1.8.2 Availability

Availability, which is the degree to which a system is ready when needed to process data, is key in providing around the clock services. The network—and particularly a network attached to a mainframe—is considered critical, and availability is mandatory for the continuity of business processes. Designers of large networks enhance availability by introducing redundant communication lines, routers, and switches, and implementing server clusters.

Maximizing redundancy has a high price tag and the network designer, together with management, must decide on the risks and impact of an outage. This will determine the availability level that suits the application and the organization. The level of reliability and redundancy introduced in mainframes is in the range of 99.999% (the five 9's), which still leaves unplanned outage of about 5.3 minutes a year. To achieve even higher availability, IBM introduced a clustering technology called Parallel Sysplex.

## 1.9 Summary

A *network* is the hardware and software that enables computers to share files and resources and exchange data. Networks play a significant role in much of the world's transaction processing. A large corporation conducts daily operations over one or more networks that connect the business—locally or remotely—to partners, suppliers, and customers around the world.

To support the changing requirements of online transactions, enterprise networks can be designed, customized, operated, and supported using the combined features and functions of network protocols, such as SNA and TCP/IP.

z/OS network capability includes a fully-featured communications server with integration of SNA and TCP/IP protocols, making the mainframe a large server capable of serving a large number of worldwide clients simultaneously.

Many technology options exist to transport, secure, protect, and encrypt z/OS hosted business sensitive and customer confidential data between the mainframe and authorized clients.

The requirements and specifications of the business transactions should determine the technologies chosen to handle the transactions.

<b>Key terms in this chapter</b>		
Advanced Peer-to-Peer Networking (APPN)	backbone	connectivity
extranet	fiber optics	infrastructure
Internet	intranet	mainframe network
network protocol	online transaction processing (OLTP)	Open Systems Adapter (OSA)
Parallel Sysplex	platform	proprietary protocol
real memory	redundancy	server farm
System Network Architecture (SNA)	TCP/IP	telephony
topology	Virtual Telecommunications Access Method (VTAM)	z/OS
z/OS Communications Server		



## Network layers and protocols review

**Objective:** As a network administrator, you must have a general knowledge of network layers, the protocols at each layer, and the hardware that facilitates the transport of data. This chapter functions as a overview for readers already familiar with IP-related layers and protocols.

After completing this chapter, you will be able to:

- ▶ Describe the network layers and their protocols.
- ▶ Describe the characteristics of local area networks and wide area networks.
- ▶ Define hubs, switches, routers, gateways, and secure gateways.
- ▶ List upper layer protocols.

If you are already comfortable with the terms associated with these topics, then you can bypass this chapter.

## 2.1 Coming to terms

It goes without saying that, between two endpoints on a network, there must be an agreement on the protocol, or language, that is in use. There is some irony in the fact that the same requirement is sometimes ignored when the communications endpoints are the network administrator and the z/OS system programmers. It is not unheard of to have the network administrator maintain that the problem is with the data link control, only to have the system programmer reply that there have been no linkage errors with any programs.

And this is a significant issue; the amount of information that network administrators and systems programmers must know in order to do their jobs is enormous. While each specialist has a clearly defined domain, some overlap is required. This chapter presents brief summaries of key terms that you need to understand in order to communicate with a network administrator.

The first sections deal entirely with IP-related networking terminology. The final section, 2.8, “Systems Network Architecture (SNA)” on page 42, offers a brief discussion of SNA. SNA topics are covered in more detail in Chapter 7, “Systems Network Architecture - basics and implementation” on page 133, Chapter 8, “SNA Advanced Peer-to-Peer Networking (APPN)” on page 165, and Chapter 9, “SNA/IP implementation” on page 211.

**Note:** It is assumed that you are already somewhat familiar with IP-related networking.

Although it is beyond the scope of this text to present IP as a new topic, there are many other sources of that information, such as the IBM Redbook *TCP/IP Tutorial and Technical Overview*, GG24-3376, available at:

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

### 2.1.1 Layers

No networking book would be complete without discussing the fact that IP networks (and SNA networks, too) are implemented as layers, as illustrated in Figure 2-1 on page 23.

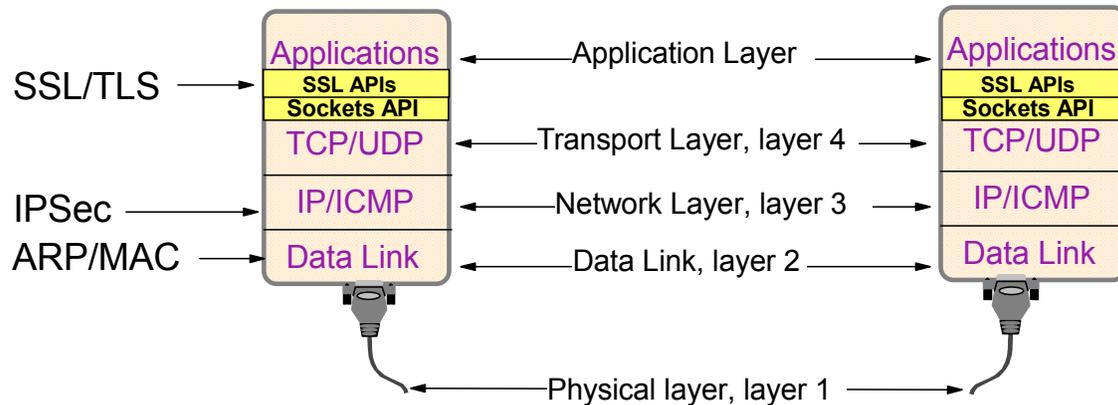


Figure 2-1 TCP/IP-based layered network

One way to look at layering in a network is as “an isolation of concerns.” Each layer has certain capabilities (in the form of protocols) that it is required to uphold. For example, the IP layer does not include reliability of delivery, while the TCP layer does not concern itself with routing details.

**Note:** Isolation of concerns (layering) of protocols is not as foreign as it first appears.

For example, when you write a check, you conform to various requirements of the check writing protocol: you must include the amount, the payee, the date and a signature on each check. When it comes to delivery of that check, you put it in an envelope and follow the postal mail protocol to ensure you have a destination address, zip code, return address and, of course, correct postage.

The postal service doesn’t care what protocol (the check writing protocol, in this case) is being encapsulated within the envelope. The content (payload) of a letter is not the concern of the mail protocol.

## 2.2 Physical media, layer 1

The physical network (also called the *physical layer*) begins at the network interface card (NIC). The NIC is effectively a method of connecting the internal data bus of a computer to the external media (cables) of the network. In the case of z/OS there is essentially only one NIC: the Open Systems Adapter (OSA) card. The OSA card is discussed in detail in Chapter 3, “Hardware connectivity on the mainframe” on page 45.

There are other ways of attaching a network to a z/OS host, but they are very seldom used. These other interfaces (parallel channel, ESCON®, and Coupling Facility attachments) are discussed in Chapter 3, “Hardware connectivity on the mainframe” on page 45.

## 2.2.1 Network interface card (NIC)

Although the OSA card is the only NIC for z/OS, this is a bit of an understatement. The OSA card variants support Ethernet in all of its current implementations. This means that it physically can connect to either a fiber optic cable or a copper (twisted pair) media. When connected to the latter, the ubiquitous RJ-45 is the connection type used.

**Note:** RJ-45 (8 wire positions) is the older sibling of RJ-11 (6 wire positions). RJ stands for Register Jack and the 11 standard is the North American (and elsewhere) standard for phone equipment—presumably everyone has seen an RJ-11 connection.

RJ-45 is the worldwide standard for copper media-based Ethernet cabling. So whether connecting a simple personal computer to a twisted pair local area network (LAN) or a enterprise scale mainframe, RJ-45 is used.

What exactly is *twisted pair*? Exactly what it says: wires running from an RJ-45 adapter are twisted as pairs and housed as a unit within a larger cable casing. This cable is referred to as UTP Cat 5, which stands for Unshielded Twisted Pair, Category 5.

The UTP Cat 5 standard tops out at 100 Mbps (megabits per second). To get faster speeds, the OSA cards switch to higher quality cabling, such as 100Base-TX. And rather than staying with copper media, higher speed networks can use fiber optic cables: 1 Gbps (gigabits per second) and 10 Gbps speeds are supported at the time of writing.

OSA card fiber optic connections can be accomplished using one of two interface types: the SC or LC. In addition, each of these interface types can be attached to one of several cable types.

Thus, in order to explain how that RJ-45 adapter attached to a UTP CAT 5 cable is going to be used, we must begin talking about *layers*. The layer that is concerned with how data signalling and movement is effected over the physical layer is called the *data link layer*.

## 2.3 Data link layer, layer 2

In Figure 2-1 on page 23, layer 2 is the data link layer. This layer is also called simply the *link layer*. The actual protocols encompassed in the link layer are numerous, and the implementation details can be found in various documents throughout the Internet and in trade texts. For the purpose of this discussion, we'll limit the scope to aspects of the link layer that a network administrator would need to know. The foremost data link layer protocol is the *Ethernet protocol*.

### 2.3.1 Ethernet

Ethernet technology is everywhere. It is believed that more than 90% of network installations use Ethernet. The remaining network connections are a combination of Token Ring, Fiber Distributed Data Interface (FDDI), Asynchronous Transfer Mode (ATM), and other protocols. Ethernet gained acceptance because of its simplicity of installation and management.

Like the check being placed into the envelope, the Ethernet protocol encapsulates data passed to it from higher layers. It also does the reverse: it decapsulates data that is presented to it from the physical layer. Thus, it stuffs envelopes when data is moving down through the layers, and it opens envelopes and passes the contents upward at the receiving end. The Ethernet envelope is called a *frame*.

The Ethernet standard was defined in 1985 by the Institute of Electrical and Electronic Engineers (IEEE) in a specification known as IEEE 802.3. The standard specifies the physical medium, carrier sense multiple access with collision detection (CSMA/CD) access method, and frame format.

In the CSMA/CD access method, each station contends for access to the shared medium. If two stations try sending the packets at the same time, a collision will result. The CSMA/CD access method is designed to restore the network to normal activity after a collision occurs, and collisions are normal in an Ethernet shared network.

The original 10 Mbps shared Ethernet network was based on coaxial cable physical medium, and later the standard was extended to shielded and unshielded twisted pair, and fiber optic cable media. The most common physical media is unshielded twisted pair (UTP), because it is inexpensive, easy to install, and allows *star topology*.

**Note:** Star topology is so named because it allows all hosts in a network to be logically (and, in effect, physically) connected at a central point. The central point of connectivity means that the loss of any individual host on the network will not affect the remaining connected hosts.

Compare this to *chain topology*, where the loss of a host in the chain would cause a disruption in connectivity.

The 10 Mbps twisted pair standard is referred to as 10Base-T.

Fast Ethernet is an extension of the popular 10Base-T Ethernet standard, supporting both 10 Mbps and 100 Mbps media speed. Fast Ethernet retains the data format and protocols of 10 Mbps Ethernet, so no changes are required in higher level protocols and applications.

Fast Ethernet standards provide for auto-negotiation of media speed, allowing vendors to provide dual-speed Ethernet interfaces that can be installed and run at either 10 or 100 Mbps. With dual speed products, users who are planning future 100 Mbps implementations can purchase a 10/100 Mbps product today and use the 10 Mbps speed in their existing networks, and then later upgrade to 100 Mbps when and where it is needed.

## Gigabit Ethernet

Gigabit Ethernet is an extension to 10 Mbps and Fast Ethernet. It provides seamless interoperability with the existing 10 Mb and Fast Ethernet (10/100 Mbps) and is compatible with existing networking protocols, networking operating systems, network applications, and networking management tools. It uses a combination of proven protocol technologies adopted by the original IEEE 802.3 Ethernet specification and Fiber channel specification.

Gigabit Ethernet retains the standard 10/100Base-T frame size and format and the same CSMA/CD scheme. However, it can use fiber channel's physical layer as the underlying transport mechanism. The full duplex implementation of Gigabit Ethernet as in Fast Ethernet does not require the CSMA/CD scheme, but retains support for the Ethernet frame format.

The initial Gigabit Ethernet offering supported one fiber physical interface. Two common fiber types in use today are *single mode fiber*, for longer distances up to 60 kilometers, and *multimode fiber* for shorter distances in the range of 300 to 500 meters. They are covered by the 1000Base-LX and the 1000Base-SX specification, respectively.

A standard has been defined by the IEEE 802.3ab task force for Gigabit Ethernet over copper physical medium.

## 10 Gigabit Ethernet

The evolution of Ethernet speeds continues. An OSA-Express2 card is also capable of supporting the 802.3 suite of standards (802.3ae) in the form of 10 Gbps. As with 1-Gigabit Ethernet, there is a copper medium option (802.3ak).

### 2.3.2 Media access control (MAC) addressing

Ethernet designates the frame format and the speed of the data travelling over the physical network. However, there is still a need for controlling how individual hosts (workstations) attached to the physical network locate each other.

The answer is the media access control (MAC) address. Every host connected to the network has a unique MAC address associated with its NIC. This MAC address, via the NIC, uniquely identifies the host.

**Note:** The address assigned to a NIC might also be referred to as a “universally administered address”, because all NICs sold worldwide (within a protocol group, such as Ethernet) must be uniquely addressed. If the address of a NIC has been manually overridden, it is considered to be a “locally administered address.”

MAC addresses are generally built into the NIC itself, but TCP/IP on z/OS does allow MAC addresses of OSA cards to be manually altered.

## 2.4 Network layer, layer 3

The most significant protocol at layer 3 (also called the *network layer*) is the Internet Protocol, or IP. IP is the standard for routing packets across interconnected networks—hence, the name *internet*. It is an encapsulating protocol similar to the way Ethernet is an encapsulating protocol. If we view the original check as a unit of data needed to be sent, we now have two envelopes required to do the transmission—the check first goes into an IP envelope, and then the entire IP envelope (known as a *packet*) is placed into an Ethernet frame.

The format of an IP packet is documented in RFC 791. The most significant aspect of the IP protocol is the addressing: every IP packet includes the IP source address (where the packet is coming from) and the IP destination address (where the packet is heading to).

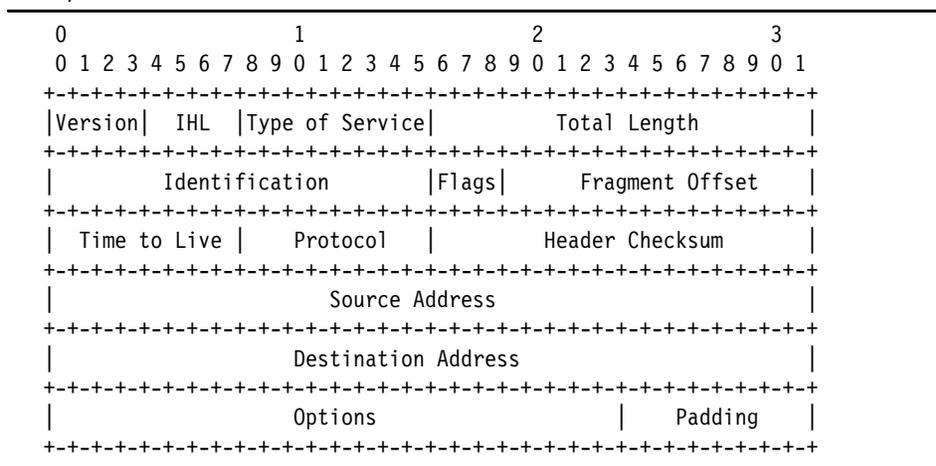
**Reminder:** What is RFC? The Internet Engineering Task Force (IETF) is an international community that keeps the world of the Internet Protocol running smoothly. The IETF governs standards for IP applications, IP-related protocols and related areas.

The standards are defined using documents called Request for Comments or RFCs. The IETF is here to stay and the RFCs they write are your friends. Get to know them at the following Web site:

<http://www.ietf.org>

Example 2-1 on page 28 shows the layout of an IP version 4 header.

*Example 2-1 IP header*



The fields that are of most interest in the example are the source and destination address fields. But the IP header also includes the Type of Service, which allows some control over the way this packet is treated as it is moved around the internet. The length of the packet must be included in this header, since IP packets can contain a variable amount of data.

Up to this point, we haven't dealt with anything other than a single network. Technically, two hosts could communicate with each other just fine using only MAC addresses and the Ethernet protocol. However, this never happens. Instead, the actual locating and delivery of data is facilitated by IP at layer 3 (see Figure 2-1 on page 23) by using layer 2. The first step for accomplishing this is the address resolution protocol, or ARP.

## 2.4.1 Address Resolution Protocol (ARP)

The Address Resolution Protocol is a layer 2 protocol used to map MAC addresses to IP addresses. All hosts on a network are located by their IP address, but NICs do not have IP addresses, they have MAC addresses. ARP is the protocol used to associate the IP address to a MAC address.

When a host wants to send a packet to another host, say IP address 10.5.5.1, on its local area network (LAN), it first sends out (broadcasts) an ARP packet. The ARP packet contains a simple question: What is the MAC address corresponding to IP address 10.5.5.1? The host that has been configured to use the IP address responds with an ARP packet containing its MAC address.

## 2.4.2 Network types

Up until now, all discussion has centered around Ethernet, which is a broadcast network type. Hence, Ethernet has the ability to do ARP broadcasts to find out what hosts are on the network.

A broadcast network has other capabilities. For example, a host can send a packet to all other hosts within the LAN segment (known as a *network broadcast*), or the host can target a subset of all other hosts on the LAN (known as a *network multicast*).

Other network types exist, such as point-to-point. A *point-to-point network*, as the name suggests, consists of only two hosts, one at each end of the network. Broadcasting is not possible or required because there is only one other host within the network.

The z/OS host supports point-to-point interfaces in various contexts. In addition, a point-to-multipoint type network is also possible, particularly in a sysplex (which is discussed in Chapter 6, “TCP/IP in a sysplex” on page 103). A point-to-multipoint network could be considered a hybrid: there are many hosts directly attached within the scope of a single network ID. However, there are no broadcast capabilities.

## 2.4.3 Local area network (LAN)

The term *local area network* is usually defined by its size: it is small and generally contained within a single room, a single building, or perhaps a small cluster of buildings. But perhaps a more accurate definition of the term LAN would be to refer to it as a physical segment within the scope of an ARP broadcast. For clarity, the term LAN segment will be used this way in the following text.

This not a cumbersome definition, however. There are tens of millions of computers attached to the network. How could an ARP broadcast possibly span all those computers every time it needs to send a packet to a particular IP address? The answer is that ARP broadcasts do not leave the physical LAN segments to which they are attached (you can read about the exceptions to this in the following note box).

Historically, a LAN segment is connected using an Ethernet hub. The hub is a layer 1 device only and thus it will repeat (transmit) any ARP packets to all hosts connected to the hub. Any network devices at the higher layers will not forward an ARP request.

**Note:** There are intelligent hubs (called *network switches*) that operate at layer 2. They may be configured as either ARP repeaters or they can cache ARP data relating to each LAN segment connected to the switch.

There are also devices called *bridges* (or *repeaters*) that operate at layer 1 and can seamlessly extend LAN segments.

## 2.4.4 Wide area network (WAN)

To summarize, at layer 3 an IP address is used to locate every host on the network. But hosts are located at layer 2 by a MAC address, not an IP address. Consequently, layer 3 uses ARP broadcasts to solicit a mapping of IP addresses to MAC addresses.

However, we have distinctly stated that the scope of an ARP address is within a LAN segment itself—unless, of course, a network switch (layer 2) or bridge (layer 1) is available to extend the scope of the segment.

For the sake of discussion, the term *wide area network* (WAN) will be used to denote a group of two or more local area networks connected at layer 3. A WAN would include the link (usually a high speed link) that is used for the interconnection of local area networks.

Other definitions of area networks exist that do not have a bearing on this text.

## 2.4.5 Virtual LAN

Although a LAN segment represents a physically contiguous network with ARP broadcast capabilities, it might also be desirable to divide such a LAN into one or more logical LANs. Such a LAN is called a *virtual LAN* (VLAN). A VLAN is implemented as an extension to the 802.3 protocol and is defined as 802.1Q.

When using 802.1Q protocol, frames leaving a host are tagged with a VLAN ID. The VLAN ID causes the packet to be recognized only by other hosts that have adapters activated to recognize that same VLAN ID. The result is that more than one VLAN can exist completely independent of each other on single physical segment. The advantage to this is that congestion on a LAN segment can be reduced and security can be improved by isolation of the traffic.

In addition, the VLAN ID can span multiple switches in a corporation. Thus, a VLAN ID can differentiate traffic across a network.

## 2.4.6 Network routing

Figure 2-2 introduces several new terms, but it also provides you with a graphical representation of some of the topics discussed earlier. This drawing is not intended to be representative of good network design. Its purpose is to continue the discussion of how layer 3 functions in an IP network.

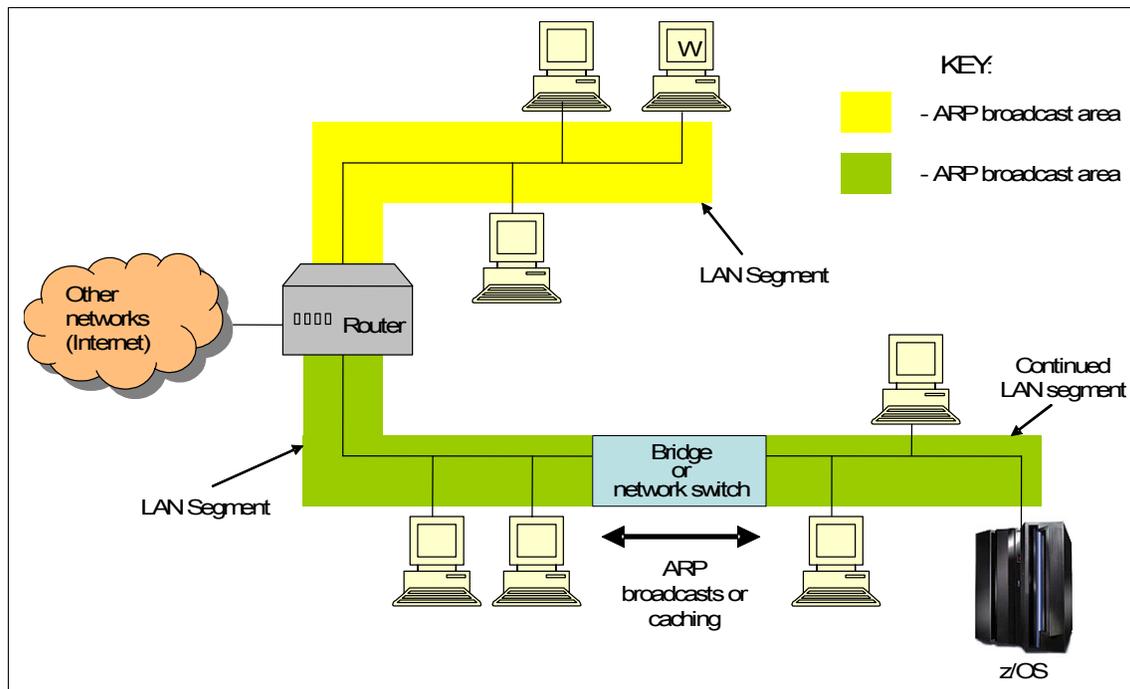


Figure 2-2 Routed network

Example 2-2 on page 33 contains two LAN segments. The upper LAN segment, in yellow, represents a simple network comprised of several workstations. The yellow area represents the scope of an ARP broadcast; the three workstations

and one of the NICs on the router are all considered to be on a single LAN segment.

The green shading on the lower half of the drawing represents a separate LAN segment. This LAN has been extended using a network switch (or a bridge) to allow separate LAN segments to function as one large, contiguous segment. In order for the router to be part of the LAN segment, it will be using a second NIC.

So how does computer W (at the top of the drawing) send an IP packet to the z/OS host on a different LAN segment at the lower corner of the drawing? The answer is that it must use an IP route to get there.

## IP routes

The IP route is the direction sign of internets, and hence of the Internet itself. An *IP route* consists of simply a mapping of a destination IP address or network to a next hop and interface. The routes are collected into a *routing table*. Each time an IP packet needs to be sent from a host, the routing table is consulted for information about where next to send the packet. To illustrate this, Example 2-2 on page 33 has been updated to contain a few IP addresses, resulting in Figure 2-3.

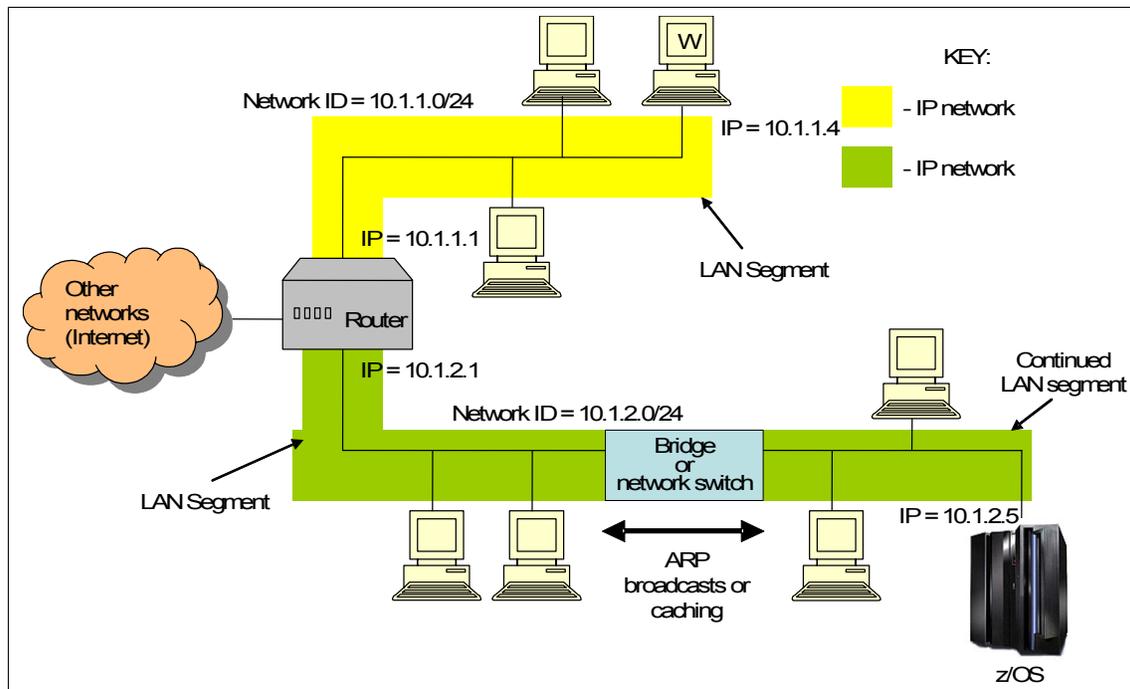


Figure 2-3 Routed network with IP addresses

Starting with a simple example, host W's routing table in Figure 2-3 would look similar to what's shown in Example 2-2 on page 33.

*Example 2-2 Sample workstation routing table*

Destination	Gateway	Interface
10.1.1.0/24	Direct	10.1.1.4
Default	10.1.1.1	10.1.1.4

In Example 2-2, the first line tells us that to reach hosts on the 10.1.1.0 network (masking with 24 bits), there is no need to use a router because the hosts for that network are directly attached to the same network as this host.

The second line says that to reach any other host, send the packet to the router at 10.1.1.1. This is referred to as a *default route*, and the assumption here is that once the packet reaches 10.1.1.1, that host will know which hop is next. (Remember, the IP layer is not reliable; it does not ensure delivery of a packet.)

**Network masks:** How did we arrive at 10.1.1.0/24 denoting a network? The 24 refers to the number of bits in the network ID portion, which corresponds to the first three octets. However, to illustrate network masking properly, a more subtle example should be used.

How about a network ID such as 201.2.10.192, with a 26-bit mask value? An octet of 255 is easy to figure out: 255 is all binary 1's, so the logical AND retains all network or subnetwork bits (we can call this the *network ID*, and ignore the arbitrary term "subnetwork"). But what about that last octet, where we are using a mask of 192?

First, we should confirm what our network ID really is:

```
Mask:           255.255.255.192 → 11111111.11111111.11111111.11000000
Address:        201.002.010.192 → 11001001.00000010.00001010.11000000
Network ID:     201.002.010.192 → 11001001.00000010.00001010.11000000
```

Now we've confirmed that the network ID in our last octet is 11000000. This means our first IP address (the beginning of our range) is 11000001, or 193 in the last octet. Our last host address (the end of our range) is 11111110, or 254 in the last octet. We cannot have a host address of all 1's, since this is reserved for broadcast, so 11111111 or 255 is excluded. The eligible IP address range is from 201.2.10.193 to 201.2.10.254.

**Tip:** You'll find this exact mask and network ID combination used in Chapter 6, "TCP/IP in a sysplex" on page 103.

Returning to Figure 2-3 on page 32, note some of the changes. The yellow and green areas are no longer labelled as ARP broadcast areas. Instead, they are referred to as IP networks. The yellow area has been given a network ID of 10.1.1.0 and the green area 10.1.2.0. Both networks have a mask value of 24 bits, or 255.255.255.0. Workstation W has an IP address of 10.1.1.4, which is within the yellow area's network ID. The z/OS host is within network ID 10.1.2.0 with IP address 10.1.2.5.

But how can we simply change an ARP broadcast area (that is, a LAN segment) into an IP network? The answer is in how the transition from layer 3 (IP layer) addressing to layer 2 (MAC in the link layer) addressing is performed.

Remember the 10.1.1.0/24 route from workstation W's routing table? This was referred to as a direct route. It is a route that informs the IP layer that IP addresses for this network ID are to be found directly attached to the LAN segment. It is here that a transition occurs; when the routing table indicates a packet should be sent to the directly-attached network, this is an indication that an ARP broadcast should be sent out to determine the MAC address of the destination host.

The result is that a given network ID must not span more than one ARP area. Once a host is reached that has a route indicating the destination is directly attached, an ARP broadcast will be sent out to request the MAC address of the destination host.

ARP responses are cached locally so that an ARP request does not need to flow every time a packet is sent to a destination host.

## 2.4.7 Building routing tables

There are two different methods for populating a routing table with routes: using static routing, or dynamic routing.

### **Static routing**

*Static routes* is the term applied to any routes in a routing table that have been manually coded (entered). For example, in Figure 2-3 on page 32, the routing requirements between networks are very simple. Routing tables such as Example 2-2 on page 33 can easily be coded directly into the host to provide all connectivity requirements.

Static routing has limitations when networks become larger. The number of routes can become difficult to manage. Also, networks can change: routers can become unavailable, causing certain routes to be unusable. At the same time, new routes can become available and these must manually be added to the

routing table before they can be utilized. To overcome such limitations, dynamic routing can be used.

## **Dynamic routing**

Dynamic routing involves the usage of routing protocols to communicate information about the status of routes and interfaces. z/OS supports two types of dynamic routing protocols: Routing Information Protocol (RIP) and Open Shortest Path First (OSPF).

### ***Routing Information Protocol (RIP)***

RIP protocol comes in two versions, RIPv1 and RIPv2. Both protocols require a server (called a *daemon*) running on the host. This daemon communicates with other hosts running an RIP daemon on the network. Information about the routing tables of each daemon host is exchanged periodically. Routing tables are built based upon information about the network supplied from other routers.

The advantage here is that if a network changes, for whatever reason, the exchange of information among routers allows this change to be communicated. The drawback of RIP is that the routing tables become large very quickly. A large network can require huge routing tables. And, RIP can be slow in recognizing changes in the network. The recognition of changes in a network by a dynamic routing protocol is referred to as *convergence*.

RIPv1 is seldom used today. RIPv2 has all the functions of RIPv1 plus some improvements. RIPv2 provides a mechanism for authentication of other routers. It provides the ability to multicast, which can reduce network traffic when compared to network broadcasting. RIPv2 also supports variable length network masks.

Most z/OS networks are moving away from using RIPv2 and are instead utilizing OSPF as the dynamic routing protocol.

RIPv2 is defined in RFC 2453.

### ***Open Shortest Path First (OSPF)***

OSPF effectively accomplishes the same thing as RIP does: it populates the routing table of a host with routes. It essentially has all the capabilities of RIPv2. However, OSPF is more scalable and configurable than RIP.

In addition, OSPF supports the organization of networks into *areas*. These areas can be used to limit the amount of information that is required to be moved around an entire internet, yet there is no compromise of connectivity.

From a network route management perspective, OSPF differs significantly from RIP. OSPF exchanges information on the state of links (interfaces) instead of routing information. Link state changes are immediately reported (using a Link

State Advertisement). Consequently, network convergence is fast and consistent. In addition, hosts participating in OSPF routing are assigned specific roles (for example, a designated router or an area border router).

The protocol itself is a state-oriented protocol. Interfaces and neighboring routers are always classified as being in a particular *state*. Events on the network will cause these states to change in a pre-determined way, providing predictability and control to the OSPF routers on the network.

The routing daemon on z/OS (called OMPROUTE) is capable of handling both OSPF and RIP interfaces concurrently.

OSPF is one of the most widely implemented routing protocols. It is defined in RFC 2328.

## 2.4.8 Internet Control Message Protocol (ICMP)

ICMP is actually a user of the IP protocol—in other words, ICMP messages must be encapsulated within IP packets. However, ICMP is implemented as part of the IP layer. So ICMP processing can be viewed as occurring parallel to, or as part of, IP processing. Therefore, in Figure 2-1 on page 23, ICMP is shown as a layer 3 protocol.

ICMP is probably most well known as the message protocol used for the `ping` command. A ping command sends an ICMP echo request to the target host. The target host responds with an echo reply. The ping command is losing some of its usefulness in today's more security-conscious networks: many routers disable responses to echo requests.

ICMP's primary use on a network is to deliver information about simple problems with the delivery of packets. For example, ICMP can inform hosts about:

### **Maximum transmission unit limitations**

When a packet that is too large for a network to handle arrives at a router, the router will break it into smaller packets (*fragments*).

If the packet has a flag (an IP flag, in fact) stipulating the packet cannot be fragmented, then the router will discard the packet and send an ICMP *fragmentation needed* packet back to the original sender.

### **Packet expiry**

The time exceeded after a packet has traversed too many hops.

**Destination unreachable**

For example, when an ARP broadcast fails to elicit a matching IP address.

**Routing problems**

When a host believes a better route exists.

Note that this is not a desirable feature of ICMP and should be disabled under almost all circumstances.

Routing protocols do a better job of determining the best route.

ICMP is defined in RFC 792.

## 2.4.9 Other layer 3 protocols

There are numerous other protocols present at the network layer. All of them are related to routing or addressing of data in some fashion or another, and usually they are more specialized with respect to their function or purpose.

## 2.5 Transport layer, layer 4

Unlike layer 3, there are really only two protocols of note found in layer 4: Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). Returning to our postal mail protocol analogy, layer 3 is preoccupied with ensuring that the address on the envelope could be located and that the envelope could ultimately be delivered. Layer 4 shifts the focus to the process of the actual delivery of the envelope.

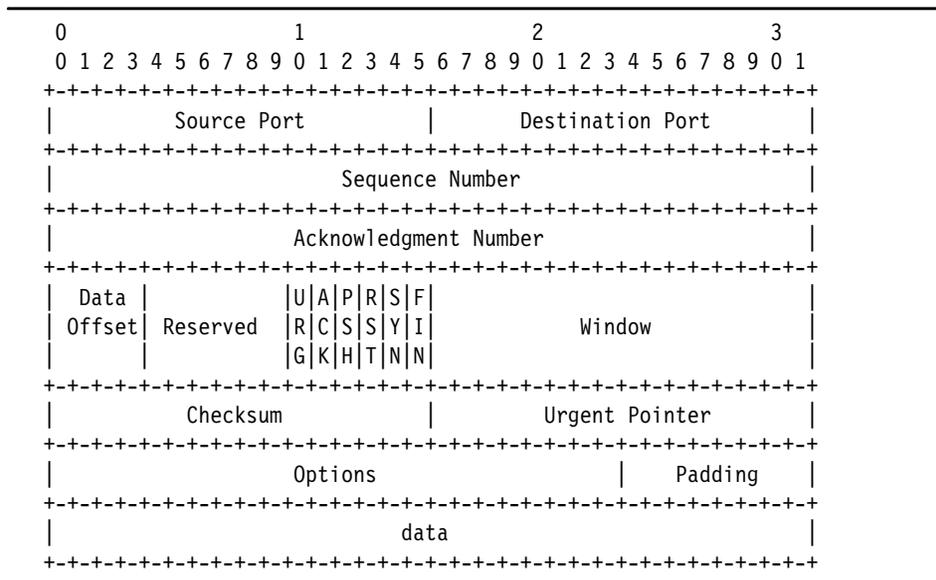
### 2.5.1 Transmission Control Protocol (TCP)

The standard way of ensuring the delivery of postal mail is to register the mail with the mail carrier. When the mail is received at the other end, an acknowledgement in the form of a signature is required. This signature is the sender's assurance that the mail has been received successfully at the remote end.

**Acknowledged data**

TCP is the "registered mail" protocol of internets. It is used when a host requires assurance that the remote end has actually received the data it sends. But instead of requesting a signature at the remote end (computers have messy handwriting anyway), TCP requires an acknowledgement be returned. To get into details on how this is done, we'll begin by having a look at Example 2-3 on page 38.

Example 2-3 TCP header



At offset 32 into the TCP header is the sequence number. The *sequence number* is a counter used to keep track of every byte sent outward by a host. If a TCP packet contains 1400 bytes of data, then the sequence number will be increased by 1400 after the packet is transmitted.

At offset 64 is the *acknowledgement number*. This number is a counter to keep track of every byte that has been received. If 1000 bytes are received by a host, it increases the acknowledgement number by 1000 when it sends out a packet in response.

**Note:** The local host's sequence number usually matches the remote host's acknowledgement number, and the local host's acknowledgement number usually matches the remote host's sequence number.

Because a transmitted packet might not reach its destination, for whatever reason, and a transmitted packet might take some time to cross the network to its destination, the difference between a sender's sequence number and the remote host's acknowledgement number represents any outstanding, unacknowledged data.

As mentioned, receiving data from a remote host causes the acknowledgement number at the local host to be increased by the number of bytes received. When the local host sends out its next packet, it will send this updated acknowledgement number, and it will also turn on the ACK flag (offset 107) to

indicate to the other end that it is acknowledging the receipt of data. This is the nearest thing to a signature that TCP can do. The result is that TCP is capable of ensuring reliable delivery of data.

## Ports

The other fields of significant note in the TCP header are the source port number and the destination port number. A TCP-capable host, and particularly z/OS, is capable of running more than one TCP application.

For example, a Web server and a FTP server might both be running on the same host, using the same IP address. After a packet has been delivered (via the NIC, and up to the IP layer), how does the host know which application should receive the packet? The answer is by using the port number.

*Port numbers* are TCP's method of knowing which application should receive a packet. Returning to our postal mail envelope, we did not mention that the address placed onto that envelope included an apartment number. Sure, the IP address gets us to the correct host, but the port number tells us *which* application on the host is the final recipient.

In order to facilitate communication, many applications are assigned specific ports. Such ports are called *well-known ports*. For example, a Web server normally listens on port 80. An FTP server normally listens on port 21.

## Connection-oriented and state-aware

TCP is always referred to as a *connection-oriented protocol*. What this entails is that prior to any communication occurring between two endpoints, a connection must be established. During the communications (which can last for seconds or for days) the state of the connection is continually tracked. And, when the connection is no longer needed, the connection must be ended.

Because TCP forms the backbone of so much activity over an internet, a summary of the possible states for a TCP connection is appropriate:

<b>LISTEN</b>	An application (such as a Web server) is awaiting an inbound connection request (from a browser, for example).
<b>SYN-SENT</b>	A connection request has been sent but no acknowledgement has been received from the remote host.
<b>SYN-RECEIVED</b>	A connection request has been received and a connection request and acknowledgement have been sent in response. Awaiting an acknowledgement of the

	connection request sent out as a response to the original connection request.
<b>ESTABLISHED</b>	All connection requests and acknowledgements have been sent and received. Data can move freely over the connection.
<b>FIN_WAIT_1</b>	An end connection request has been sent, but no acknowledgement has been received.
<b>FIN_WAIT_2</b>	An end connection request has been acknowledged by the remote host, but no corresponding end connection has been received from the remote host.
<b>CLOSING</b>	An end connection request has been sent out and an end connection response has been received and acknowledged. However, the remote end has not yet acknowledged the original end connection request.
<b>CLOSE_WAIT</b>	An end connection request was received and acknowledged but no corresponding end connection has been sent out yet.
<b>TIME_WAIT</b>	Waiting a reasonable amount of time to ensure that the final acknowledgement of a received end connection has been received at the remote end.
<b>LAST_ACK</b>	Awaiting a final acknowledgment after sending an end of connection in response to having received an end of connection request.

## 2.5.2 User Datagram Protocol (UDP)

When postal mail is not registered, there is a small chance that the letter we send might never be seen again. The letter is addressed and sent and no more effort is spent on it. It is a matter of the mail carrier doing its job correctly.

It is the same case with UDP. A UDP header containing an IP address and a port number is wrapped around whatever data needs to be sent, and the packet is handed over to the IP layer. As long as the lower layers do their jobs correctly, the remote end should receive the datagram as expected. There are no acknowledgement counters and no connection states.

## 2.5.3 Sockets

The term *socket* in a TCP or UDP context fully describes the endpoint of a connection. The socket is consequently a combination of an IP address, a port number, and the protocol being used.

## 2.6 Applications

Sitting above layer 4 are the applications. Applications can use either TCP or UDP to communicate. Because of its inherent reliability, TCP tends to be used more often. Examples of applications running on z/OS using TCP include sendmail, Web servers, FTP and telnet. Applications using UDP on z/OS are Traceroute, Enterprise Extender (discussed in 9.3, “Enterprise Extender” on page 220), and name servers (Domain Name System).

**Tip:** Enterprise Extender is really SNA encapsulated in a UDP datagram. At first, it appears odd that UDP and not TCP would be chosen to carry traffic as important as SNA traffic. However, the flow control and reliability capabilities of TCP are already built into SNA, so TCP is not required.

## 2.7 Security

Security is never far from the mind of any administrator of a network. Although security is covered extensively in a z/OS context in Chapter 12, “Network security” on page 283, it is useful to clarify some security terminology here.

### 2.7.1 Firewalls and gateways

Firewalls are so common that a definition is hardly needed; however, in a large organization the term should be formally defined.

A firewall is an implementation (or extension) of an organization’s security policies. Any large organization has (or should have) a formal document explaining the classification of company data, as well as the classification of company networks.

A firewall controls and limits access between networks of different security classifications, and sometimes even within a network that is already protected by a firewall. Firewalls can filter based upon port numbers and IP addresses (or networks).

Firewalls also often function as endpoints for secure communications across a non-secure network. Data travelling from the secure network outward will be secured as it crosses the non-secure network (a requirement of the organization’s security policy, no doubt). Data travelling into this firewall would likewise be secured.

Generally, a firewall acting in this fashion is running an IPsec protocol (RFC 2401 through RFC 2409) in the form of a virtual private network (VPN). We discuss this topic in more detail in Chapter 12, “Network security” on page 283.

A firewall that acts as a VPN endpoint and allows data to continue on through the secure network to destination hosts is often called a *security gateway*. The term gateway is traditionally used to describe a host that connects networks using different protocols.

## 2.7.2 Security protocols

Looking at Figure 2-1 on page 23, the IPsec protocol is implemented at the network layer. An alternative form of security for data on the network is the Secure Sockets Layer (SSL). SSL is implemented at the transport layer. The new standard for SSL is called Transport Layer Security, and is also discussed further in Chapter 12, “Network security” on page 283.

## 2.7.3 Protection

A host such as z/OS includes intrusion detection services (IDS) that allow the host to detect and react to malicious activities coming from the network. Some IDS is built into TCP/IP on z/OS itself, while other aspects of IDS are configurable. IDS can be an integral part of host availability.

## 2.8 Systems Network Architecture (SNA)

“Classic” SNA, based on subarea nodes, is the original networking architecture used by mainframe computers. However, with the popularity and growth of TCP/IP, SNA is changing from being a true network architecture to being what could be termed an “application and application access architecture.” In other words, there are many applications that still need to communicate in SNA, but the required SNA protocols are carried over the network by IP.

“New” SNA is SNA/APPN (Advanced Peer-to-Peer Networking). This addition to SNA is based on APPN Peer nodes, and is somewhat more dynamic and less deterministic than SNA/Subarea, and usually requires considerably less definition. It is possible to have SNA/APPN traffic travel over an Ethernet LAN network (and even between LANs) using the IEEE 802.2 frame format. The 802.2 frame is referred to as *logical link control* (LLC).

Similar to an 802.3 frame, the 802.2 frame contains something called a *service access point*, or SAP. The SAP is used to identify the SNA resource at the receiving host.

SNA is discussed in detail beginning in Chapter 7, “Systems Network Architecture - basics and implementation” on page 133.

## 2.9 Summary

The first step in discussing network technology is to ensure that you understand the terms and acronyms. Starting from the physical layer, progressing to the data link layer (Ethernet), and moving up through the network layer (IP and routing) on to the transport layer (TCP and UDP), there are a large number of terms to be understood. These terms need to be clearly understood when z/OS systems programmers communicate with network administrators in an organization.

<b>Key terms in this chapter</b>		
Address Resolution Protocol (ARP)	firewall	frame
gateway	hub	Internet Control Message Protocol (ICMP)
Internet Protocol (IP)	intrusion detection services (IDS)	IPsec
local area network (LAN)	MAC address	network interface card (NIC)
Open Systems Adapter (OSA)	packet	port
Request For Comments (RFC)	Routing Information Protocol (RIP)	service access point (SAP)
socket	Systems Network Architecture protocol	Transmission Control Protocol (TCP)
User Datagram Protocol (UDP)	virtual private network (VPN)	wide area network (WAN)





## Hardware connectivity on the mainframe

**Objective:** This chapter describes the various ways a mainframe connects to a network.

After completing this chapter, you will be able to:

- ▶ Understand the different types of hardware connections to a network.
- ▶ Understand the protocols running over these different connections.
- ▶ Understand the configuration options and advantages of OSA-Express.
- ▶ Explain how to associate protocols with the OSA-Express.

## 3.1 Connecting the mainframe

The design intention of the mainframe, and most of its evolution, is for the mainframe to be a highly available transaction processing server. Obviously, central processing capabilities are evolving to handle more and more transactions. However, in order to be an effective transaction processing server, there must be a proportional capability of moving data in and out of the central processor complex rapidly (CPC, the physical collection of hardware that consists of main storage, one or more central processors, timers, and channels). The result is that the I/O (input/output) options, capabilities and configuration choices of an IBM mainframe are varied, complex, and very performance oriented.

Mainframe computers are probably unique in that they require a Hardware Management Console, or HMC. The HMC is a separate interface to the central processor complex that is used for hardware configuration operations. It also provides an interface to the z/OS system console.

## 3.2 Channel subsystem (CSS)

The heart of moving data into and out of a mainframe host is the *channel subsystem*, or CSS. The CSS is, from a central processor standpoint, independent of the processors of the mainframe host itself. This means that input/output (I/O) within a mainframe host can be done asynchronously. When an I/O operation is required, the CSS is passed the request from the main processor. While awaiting completion of an I/O request, the main processor is able to continue processing other work. This is a critical requirement in a system designed to handle massive numbers of concurrent transactions.

**Note:** The processors that run the channel subsystem are called the *system assist processors* (SAP). There can be more than one SAP running the channel subsystem.

All LPARs within the central processor complex can make use of the channel subsystem.

**Reminder:** A *logical partition*, or LPAR, is an independent subset of a CPC. Operating systems and applications running within an LPAR cannot distinguish the LPAR resources (memory, processors and devices) from those of a dedicated CPC. Effectively, an LPAR is a seamless division of a CPC into multiple simulated CPCs.

A simplified example of how the channel subsystem functionally resides within a central processor complex is shown in Figure 3-1 on page 47. In this diagram, the large box (in pale blue) represents an entire mainframe processor (CPC).

The asynchronous I/O is handled within the channel subsystem by a channel program. Each LPAR ultimately communicates using a subchannel. In addition, the channel subsystem can be used to communicate between LPARs.

Each CPC has a channel subsystem. Its role is to control communication of internal and external channels to control units and devices.

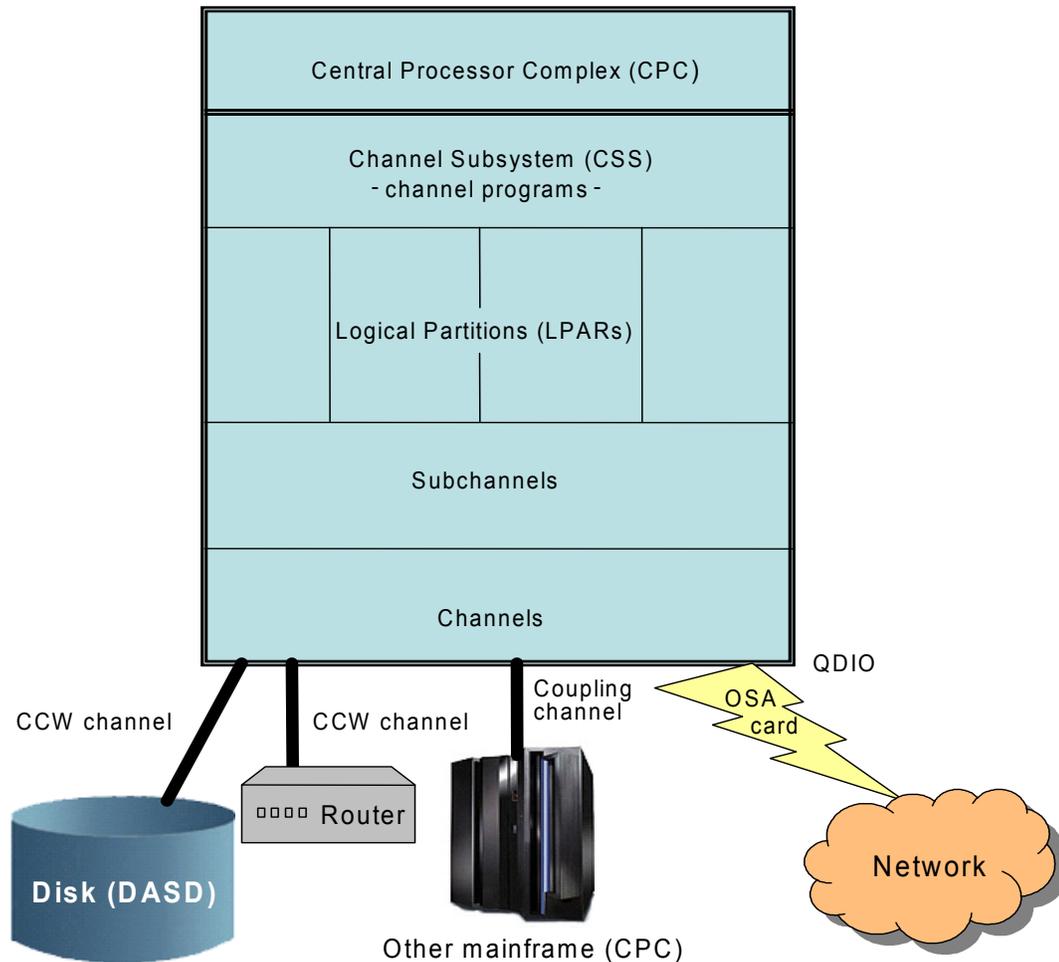


Figure 3-1 I/O and the mainframe

The channels permit transfer of data between main storage and I/O devices or other servers under the control of a channel program. Some of the other components in Figure 3-1 are described as follows:

<b>Partition</b>	Within the central processor complex (CPC) are logical partitions that divide the CPC into independent machines that can run any mainframe architecture system control program (for example, z/OS, Linux, or z/VM). Partitions have access to CPC memory and subchannels.
<b>Subchannel</b>	The subchannel represents an I/O device. This is the mechanism by which an I/O request is passed (identified) to the channel subsystem itself.
<b>Channel</b>	<p>The channel, represented by a channel path ID or CHPID, represents the actual communication path. A CHPID is the handle by which communication between the CPC and an external device is facilitated.</p> <p>A CHPID must be unique, since it denotes a unique path of communication for the CPC. The maximum number of allowable CHPIDs within a channel subsystem is 256. Channels can be shared between LPARs.</p> <p>Historically, a CHPID had a correspondence with a real physical channel connected to the CPC. However, for performance and enhanced capabilities, a CHPID now maps to a physical CHPID (PCHID) using a simple mapping table and a CHPID mapping tool, or CMT.</p>

### 3.2.1 Control units

One of the main tasks of the channel subsystem is to communicate with storage devices such as tape and direct access storage devices (DASD). This is facilitated by a control unit (which is not shown in Figure 3-1 on page 47). Although this is a significant aspect of the channel subsystem, this will not be discussed within this text since it is not a network device.

### 3.2.2 Logical channel subsystem (LCSS)

To facilitate the usage of more CHPIDs, the mainframe architecture supports a logical channel subsystem, or LCSS. The LCSS is functionally identical to the channel subsystem, but up to four LCSSs can be defined within a central processor complex. CHPIDs are unique within the LCSS only; consequently, the 256 CHPID limitation can be overcome.

**Note:** The I/O configuration of the central processor complex is defined in a data set called the I/O Configuration Data Set, or IOCDs. The I/O configuration is normally done using a tool called the Hardware Configuration Dialog, or HCD. HCD also creates a data set called an I/O definition file, or IODF. The IODF is read by the z/OS operating system.

A central processor complex can also be configured using a less easy-to-use statement syntax called IOCP statements. IOCP stands for I/O Configuration Program (IOCP). The IOCP creates an I/O configuration data set (IOCDs). IOCP statements can be migrated to IODF statements using HCD.

### 3.3 The mainframe channel subsystem and network links

As mentioned, a CHPID no longer directly corresponds to a hardware channel, and CHPID numbers may be arbitrarily assigned. A hardware channel is now identified by a physical channel identifier, or PCHID.

A z990 with multiple logical channel subsystems is shown in Figure 3-2, which gives us an opportunity to go a little deeper into how the I/O subsystem functions.

Two logical channel subsystems are defined (LCSS0 & LCSS1). Each LCSS has three logical partitions with their associated MIF identifiers. An explanation of MIF and the other features within Figure 3-2 follows.

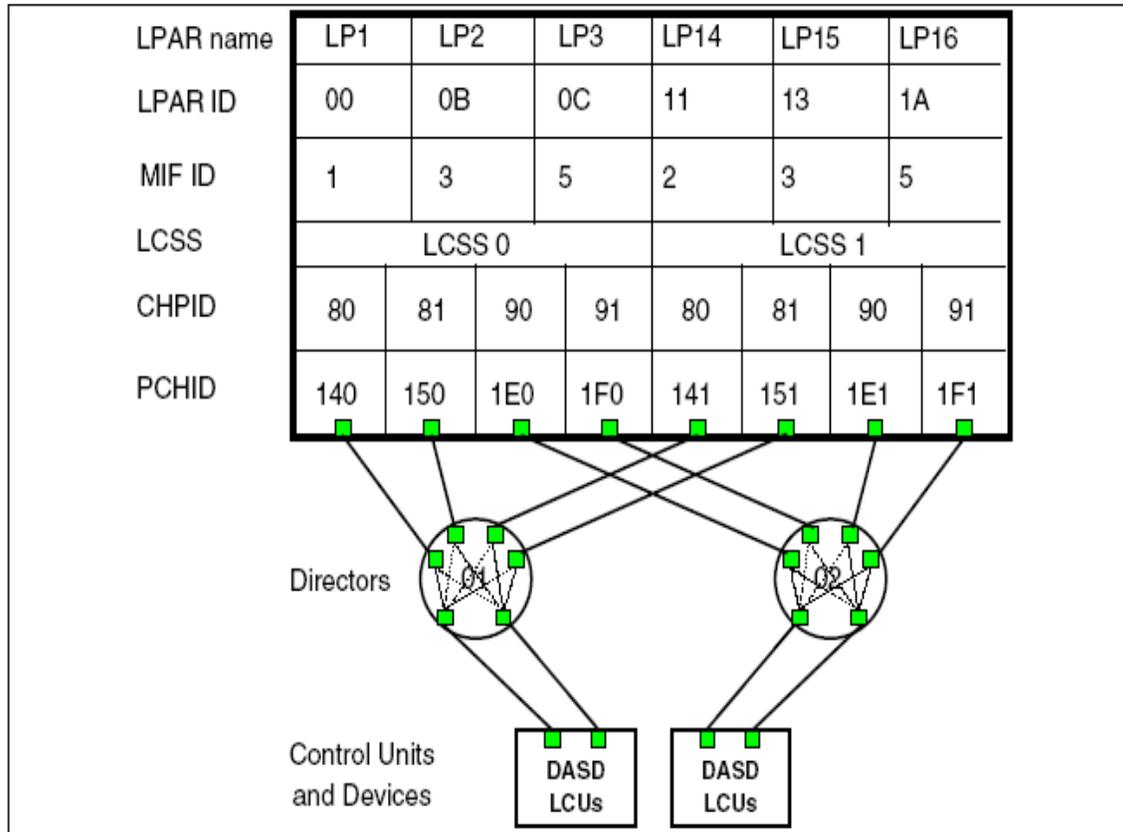


Figure 3-2 Logical channel subsystem (LCSS) connectivity

**LPAR name (logical partition name)**

This name is user-defined through HCD or IOCP and is the partition name in the RESOURCE statement in the configuration definitions.

The names must be unique across all logical channel subsystems defined for the z990.

**LPAR id (logical partition identifier)**

The logical partition identifier is a number in the range from '00' to '3F'. It is assigned by the user on the image profile through the support element (SE) or the Hardware Management Console (HMC).

**Note:** The logical partition identifier is unique within the central processor complex.

### ***MIF ID (multiple image facility identifier)***

The MIF number is used to facilitate channel sharing among LPARs.

The MIF ID is a number that is defined through Hardware Configuration Dialog (HCD). It is a number that is specified in the RESOURCE statement in the configuration definitions.

It is in the range '1' to 'F' and is unique within a logical channel subsystem, but it is not unique within the z990. Multiple logical channel subsystems may specify the same MIF ID.

### ***CHPID (channel path identifier)***

CHPID number is associated with a physical channel port location (PCHID), and a logical channel subsystem. The CHPID number range is still from '00' to 'FF' and must be unique within a logical channel subsystem.

### ***Control unit***

The CU provides the logical capability necessary to operate and control an I/O device, and it adapts the characteristics of each I/O device to the standard form of control provided by the channel.

A control unit may be housed separately, or it may be physically and logically integrated with the I/O device. In Figure 3-2 on page 50, the control units have been logically divided into logical control units, or LCUs.

### ***I/O device***

I/O devices are used to provide external storage (for example, disk storage called DASD or direct access storage devices), to communicate between data processing systems, and to communicate between a data processing system and the external world.

### ***Director***

A director is an I/O interface providing multiple connectivity capabilities between the channels on the mainframe and the control units of the devices.

## **3.4 Hardware channels**

This topic describes the different channels on a mainframe and how they attach to the network. How the mainframe connects to the network depends on the channel type used.

There are effectively three ways that network traffic can travel between an external network and a z/OS host: through a channel-command word channel, a coupling channel, or a QDIO channel.

## 3.5 Channel command word (CCW)

The CCW is the original I/O operation used for communications with the channel subsystem. The CCW contains a channel command, such as read, write, or control, along with the data address of the data area involved. The data is passed to the channel subsystem. The channel subsystem communicates status of the I/O back to the issuing application. When a channel communicates with an application in an asynchronous fashion, it is referred to as a *channel interrupt*.

### 3.5.1 CCW-based operations

CCW-based channels include parallel, ESCON, and FICON channels. A CCW can also be used to talk to an OSA card (this is discussed further in 3.7, “Open Systems Adapter (OSA)” on page 56).

### 3.5.2 Parallel channel

The parallel channel is one of the oldest interfaces to a mainframe. It was introduced to the mainframe in the 1960s. There may be some still in use today, but newer mainframes no longer support parallel channels.

The parallel channel (also called an S/370™ I/O interface) uses two cables called *bus* and *tag*. Of course, it is copper media only. A bus cable carries information (one byte in each direction), and a tag cable indicates the meaning of the information on the bus cable. Devices are daisy-chained off of each other to form a string of devices.

The parallel I/O interface is the communication channel path between a physical channel on a mainframe and an I/O control unit (CU). The interface was designed for past mainframes and works compatibly with older and current mainframe architectures.

**Note:** Some new mainframe models, such as the z/890, z/990, and z/9-109 do not support parallel channels.

The maximum data rate of a parallel channel is 4.5 MBps (megabytes per second).

### 3.5.3 Enterprise System Connectivity (ESCON) channel

ESCON replaces the previous S/370 parallel channel with the ESCON I/O interface, supporting additional media and interface protocols. By replacing the previous bus and tag cables and their multiple data and control lines, ESCON provides half-duplex serial bit transmission. In contrast to the previous copper cables used in the parallel channel, ESCON utilizes fiber optic cables for data transmission.

**Reminder:** Half-duplex for ESCON is effectively a request-response format. Bi-directional communications are possible, but the synchronization of the ESCON I/O protocol limits communications to half-duplex.

An ESCON channel executes commands presented by the standard z/Architecture™ or ESA/390 I/O command set, and it manages its associated link interface (link level/device level) to control bit transmission and reception over the physical medium.

See Figure 3-3 on page 54 for an illustration of channel operation data flow. At the top layer, an application makes an I/O request using a macro or a supervisor call (SVC). This in turn causes a START SUBCHANNEL (SSCH), which moves the I/O request to the CSS. The SSCH includes a subsystem identifier (SSID) and operation-request block (ORB) as its operand for execution of the channel program.

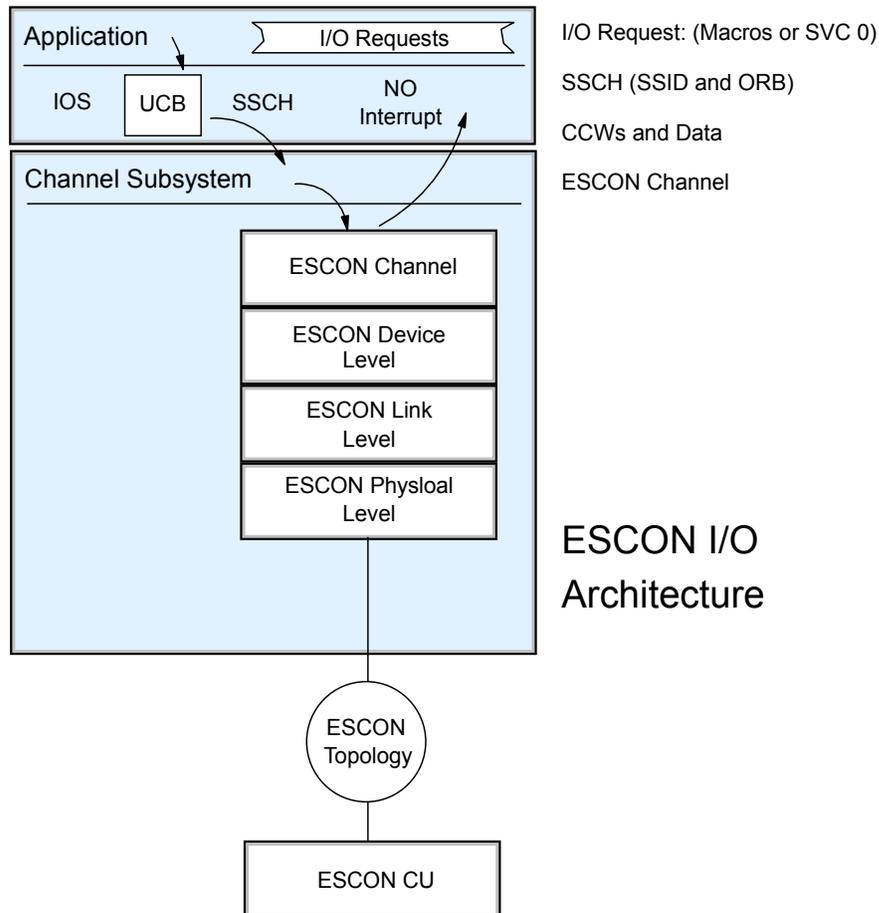


Figure 3-3 ESCON channel operation flow

ESCON has a somewhat different topology for control unit and channel attachment compared to a parallel channel. ESCON control units can be connected:

- ▶ Directly to an ESCON channel, which is called point-to-point, or
- ▶ Dynamically switched through a device called the ESCON Director, which is called switched point-to-point

In order to accommodate parallel channel-attached control units and devices, the ESCON conversion mode allows communication from an ESCON channel to parallel channel-attached control units.

The maximum channel data rate of an ESCON channel is 17 MBps (megabytes per second) and the maximum unrepeated distance is 3 kilometers.

### 3.5.4 Fiber connection (FICON)

Even though ESCON channels are fiber-based, the next generation of ESCON was simply called FICON, for fiber connection. The advent of FICON allowed concurrent sharing of the fiber channel (up to 8 operations at the same time). Other advantages included:

- ▶ The maximum channel data rate of 2 Gbps (gigibits per second)
- ▶ Up to 10 km for an unrepeated distance
- ▶ More device numbers supported

### 3.5.5 Channel-attached network devices

At this stage, we have discussed in detail channel connection types, but what is being attached at the other end of the channel? As mentioned earlier, it is often a control unit with DASDs attached. In a networking context, however, the device attached at the other end can be a router.

#### **Routers**

In order to have z/OS talk to a channel-attached router, a protocol above the channel protocol must be agreed upon. The protocol used for this is called CLAW. CLAW stands for Common Link Access to Workstation. CLAW can be used to talk to either a CISCO CIP (Channel Interface Processor) host or an AIX pSeries® host. CLAW-connected hosts are steadily becoming less common in z/OS networks.

#### **Other LPARs**

The parallel channel can of course be used to connect two LPARs directly, or even two separate central processor complexes. The other LPAR could be running z/OS, or it could be running z/VM with multiple Linux images within a single LPAR.

## 3.6 Coupling channels

Communication among LPARs can be facilitated by Coupling Facility (CF) links. Coupling Facility links are used to support the cross-system Coupling Facility, or XCF. The XCF component in turn can be used to support the IP protocol. There are two ways to define IP connectivity over a Coupling Facility link:

<b>Static XCF</b>	Static XCF links can be defined to TCP/IP on z/OS using hardcoded statements.
<b>Dynamic XCF</b>	Dynamic XCF links can be automatically generated any time TCP/IP becomes active within a sysplex.

**Note:** z/OS hosts interconnected using XCF are said to be *tightly coupled hosts*. In z/OS, a group of tightly coupled hosts are referred to as a *sysplex*.

XCF communications can flow over copper or fiber media. Over fiber media, the maximum data rate is 200 MBps. Additionally, coupling channels are not CCW-based channels.

## 3.7 Open Systems Adapter (OSA)

The Open Systems Adapter is actually a network controller that you can install in a mainframe I/O cage. The adapter integrates several hardware features and supports many networking transport protocols. The OSA card is the strategic communications device for the mainframe architecture. It has several key features that distinguish it from CCW-based communications.

Effectively, the OSA integrates the control unit and device into the same hardware. It does so by placing it on a single card that directly connects to the central processor complex I/O bus.

There are three main versions of the Open Systems Adapter:

- ▶ The OSA-2
- ▶ The OSA-Express
- ▶ The OSA-Express2

The OSA-2 card is no longer available, but a significant number of installations still utilize them. The OSA-2 card is of interest here because it could only run using CCW-based operations. OSA-Express and OSA-Express2 cards utilized a much faster method of direct access called Queued Direct I/O (QDIO).

In addition, OSA-Express provides significant enhancements over the OSA-2 in function, connectivity, bandwidth, data throughput, network availability, reliability, and recovery. Meanwhile, the OSA-Express2 card represents the latest and most capable card in the OSA lineup.

Figure 3-4 shows the OSA-Express2 and OSA-Express features that are available on mainframe servers.

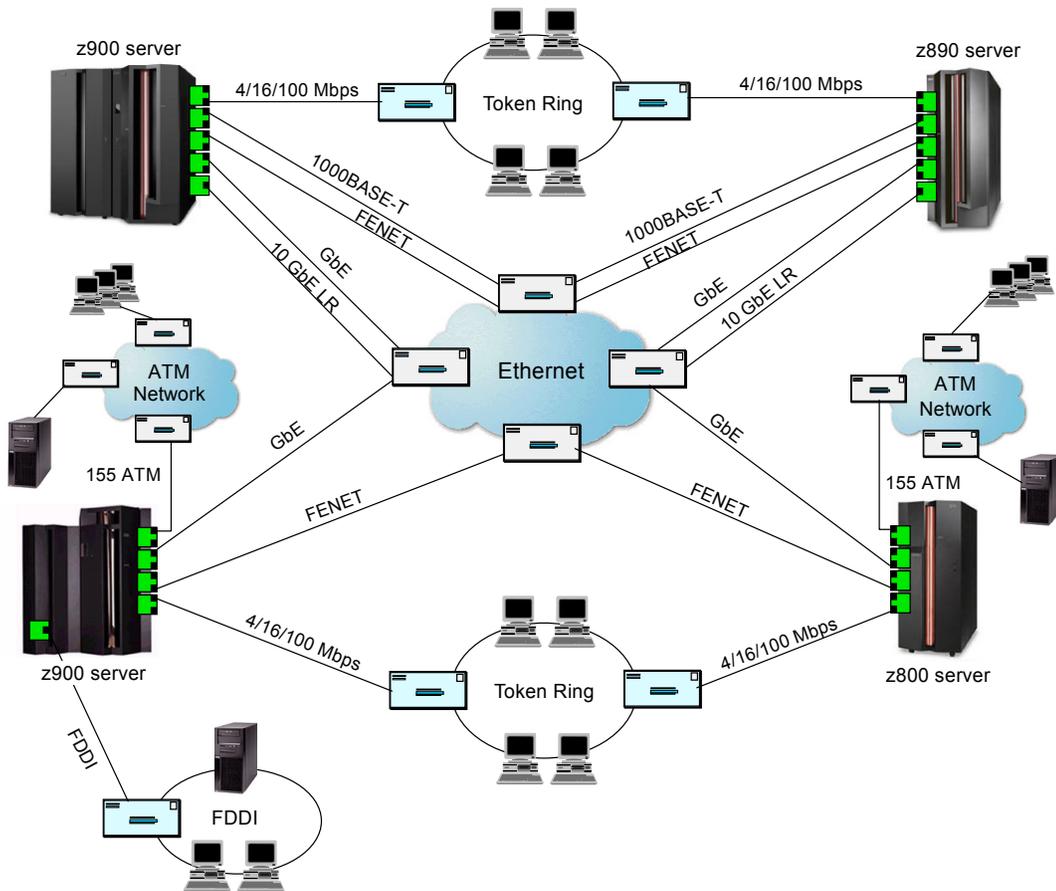


Figure 3-4 OSA-Express and OSA-Express2 connectivity

Note that the maximum speed is a 10 Gbps data rate. In order to support such a large potential for data movement, as mentioned the OSA-Express and OSA-Express2 cards support a mode of operation called Queued Direct I/O, or QDIO. There are several different channel types supported by an OSA-Express2 card:

- OSD** Queued Direct I/O (QDIO)
- OSE** Non-Queued Direct I/O (non-QDIO)
- OSC** OSA-Express Integrated Console Controller
- OSN** Network Control Program (NCP) under Communication Controller for Linux (CCL).

Only the OSA-Express2 card supports the OSC and OSN channel types. The following text uses the term “OSA-Express” to denote a function that both OSA-Express and OSA-Express2 can support.

### 3.7.1 Queued Direct I/O (QDIO)

QDIO mode is referred to as OSD because the CHPID type coded in the IOCDS is OSD.

Figure 3-5 on page 59 illustrates the much shorter I/O process when in QDIO mode compared with non-QDIO mode (the same I/O path as the OSA-2 features). Consequently, I/O interruptions and I/O path-lengths are minimized.

The advantages of using QDIO mode are:

- ▶ A 20% improvement in performance versus non-QDIO mode
- ▶ The reduction of system assist processor (SAP) utilization
- ▶ Improved response time
- ▶ Server cycle reduction

How is this all accomplished? Instead of attaching an OSA-Express card using a channel, and hence utilizing a channel or I/O program (IOP) combined with CCW operations, the OSA-Express card attaches using an STI bus. STI stands for Self-Timed Interface. The OSA-Express card is still connected within the I/O cage, but the STI bus is directly connected to the memory bus of the CPC.

What is used instead of a CCW operation to signify that I/O needs to be completed? With an OSA-Express card running in QDIO mode, I/O operations are effected using a signal adapter instruction, or SIGA. The SIGA is still processed by the SAP, similar to the way a CCW is processed by the SAP. However, the SIGA effectively passes a pointer to the data because the data already occupies internal storage.

This bus itself has a data speed of up to 2.7 GBps (gigabytes per second), which is plenty of bandwidth to handle a 10 Gbps Ethernet LAN speed.

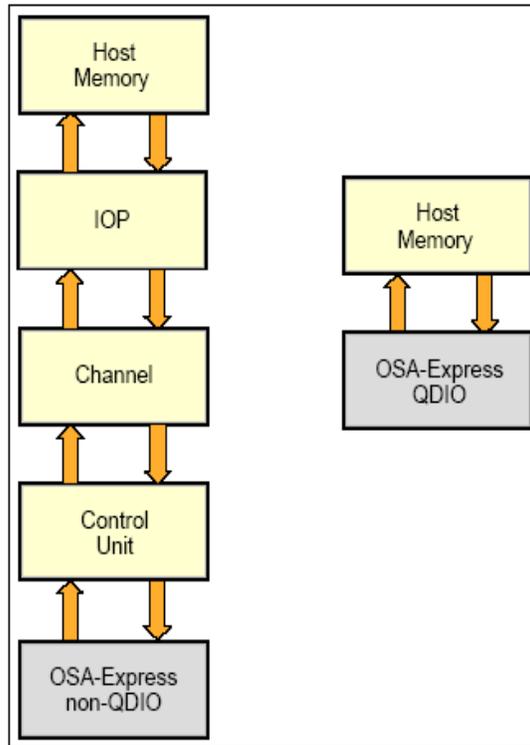


Figure 3-5 Non-QDIO versus QDIO data paths

QDIO is a highly efficient data transfer architecture, which dramatically improves data transfer speed and efficiency for TCP/IP traffic.

QDIO incorporates a number of features:

- ▶ LPAR-to-LPAR communication
 

Access to an OSA-Express port can be shared among the system images that are running in the LPARs to which the channel path is defined to be shared. Also, access to a port can be shared concurrently among TCP/IP stacks in the same LPAR, in different LPARs, or in different logical channel subsystems. When port sharing, an OSA-Express port operating in QDIO mode has the ability to send and receive IP traffic between LPARs without sending the IP packets out to the LAN and then back to the destination LPAR.
- ▶ DMA (direct memory access)
 

DMA allows data to move directly from the OSA-Express microprocessor to the host memory. This bypasses three layers of processing that are required when using ESCON and OSA-2 features, dramatically improving throughput.

- ▶ Priority queuing

Priority queuing (for z/OS environments) sorts outgoing IP message traffic according to the priority assigned in the IP header (using the Type Of Service field). This priority is used to reflect the business priorities assigned to the application, user ID, time of day, and other characteristics.

- ▶ Enhanced IP network availability

Enhanced IP network availability (IPA) is a service of the QDIO architecture. When TCP/IP is started in QDIO mode, it downloads all the home IP addresses in the stack and stores them in the OSA-Express feature. The OSA-Express feature port then responds to ARP requests for its own IP address, as well as for other IP addresses active in the TCP/IP stack (in particular with virtual IP addresses (VIPA), discussed in Chapter 5, “TCP/IP on z/OS” on page 81).

- ▶ VLAN support

IEEE standard 802.1Q describes the operation of virtual bridged LANs, known as VLANs. A VLAN is defined to be a subset of the active topology of a LAN. The OSA-Express features provide for the setting of multiple unique VLAN IDs per QDIO data device. They also provide for both tagged and untagged frames to flow from an OSA-Express port.

Full VLAN support is offered for all OSA-Express Ethernet features available on mainframe servers. z/OS Communications Server versions 1.5 and later support VLAN identifications (VLAN IDs).

**Reminder:** A VLAN frame looks almost the same as an Ethernet frame. The difference is that a VLAN frame has an extra field containing a number that identifies the VLAN. This number is called a VLAN tag.

- ▶ ARP Takeover

The Address Resolution Protocol (ARP) Takeover provides the capability of switching OSA-Express port operations from one OSA-Express to another OSA-Express running in the same mode.

When TCP/IP is started in QDIO mode, it downloads all the home IP addresses in the stack and stores them in each OSA-Express feature to which it has a connection. This is a service of QDIO architecture and occurs automatically only for OSD channels.

If an OSA-Express feature fails while there is a backup OSA-Express available on the same network ID, TCP/IP informs the backup OSA of which IP addresses (real and VIPA) to take over, and the network connection is maintained. The takeover is effected by something called a gratuitous ARP. A *gratuitous* ARP is an unsolicited ARP response. All hosts on the LAN

segment that receive this gratuitous ARP will update their ARP cache with the new MAC address for the backup OSA.

### 3.7.2 Non-QDIO mode

When the CHPID type is set to OSE, the OSA-Express card is functioning in non-QDIO mode. An OSE channel type does not support the many of the features of an OSA-Express running QDIO mode. For example, direct memory access and enhanced IP availability are only available with a channel type of OSD.

So, why run non-QDIO mode at all? In non-QDIO mode, an OSA-Express card can support SNA and APPN traffic (using 802.2 frames). In addition, in OSE mode, an OSA-Express card can run IP and SNA/APPN traffic concurrently. Some manual configuration is required, using a program called the OSA Support Facility, or OSA/SF.

### 3.7.3 OSC mode

The OSC CHPID type is available on newer mainframes running an OSA-Express2 card or an OSA-Express card with the Gigabit Ethernet feature. The OSC is a special channel type that eliminates the need for an external console controller. The end effect is that access to the HMC and to the z/OS system console is made easier. The OSC CHPID can also be used to connect TN3270 sessions (with some limitations).

**Note:** TN3270 is effectively an SNA-based telnet protocol. It is the fundamental connectivity method for interacting with z/OS. It is discussed in detail in Chapter 10, “TN3270 Enhanced” on page 241.

Because the OSC mode is not a general usage mode of operation, no further discussion of it is undertaken here.

### 3.7.4 Open Systems Adapter for NCP (OSN)

The OSN type is only available with OSA-Express2 and requires a z9 mainframe or later model. The primary intention of this type is to free organizations from the constraints of obsolete hardware: device types 3745 and 3746. The 374x device types, as they are called, are no longer manufactured or sold by IBM. A 374x host is required to run the Network Control Program (NCP). NCP is a significant functional component of subarea type SNA networks (more information about SNA networks is covered in Chapter 7, “Systems Network Architecture - basics and implementation” on page 133).

The OSN channel type allows an Open Systems Adapter to communicate with an NCP using Channel Data Link Control protocol (CDLC). CDLC cannot be used over an OSD or OSE channel type, and even with channel type OSN it can only communicate to other LPARs within the CPC. Historically, 374x devices were often connected to a parallel or ESCON channels, which support CDLC.

Where, then, will the NCP run? On a software program called Communications Controller for Linux (CCL). And, as mentioned, both LPARs must be within the same CPC, since the data flows do not enter the network.

In many cases, CCL provides the easiest way to migrate from older SNA-based network controllers to modern network devices. The CCL functional capabilities provide alternatives where either no viable solution existed before (XRF, for example) or where prior alternatives (such as SNI) posed significant implementation challenges.

**Clarification:** XRF is the Extended Recover Facility. XRF is a feature of SNA/APPN environments that greatly improves the recoverability of an application or host failure.

SNI stands for SNA Network Interconnect. SNI is used when connecting two separate subarea SNA networks to each other.

Figure 3-6 on page 63 shows a connectivity example. In this example, a channel type of OSE is used to communicate with CCL using LLC 802.2. With the OSN channel type, the communication would not flow out to the switch. Instead, CDLC would flow from LPAR to LPAR within the OSA-Express2 card itself.

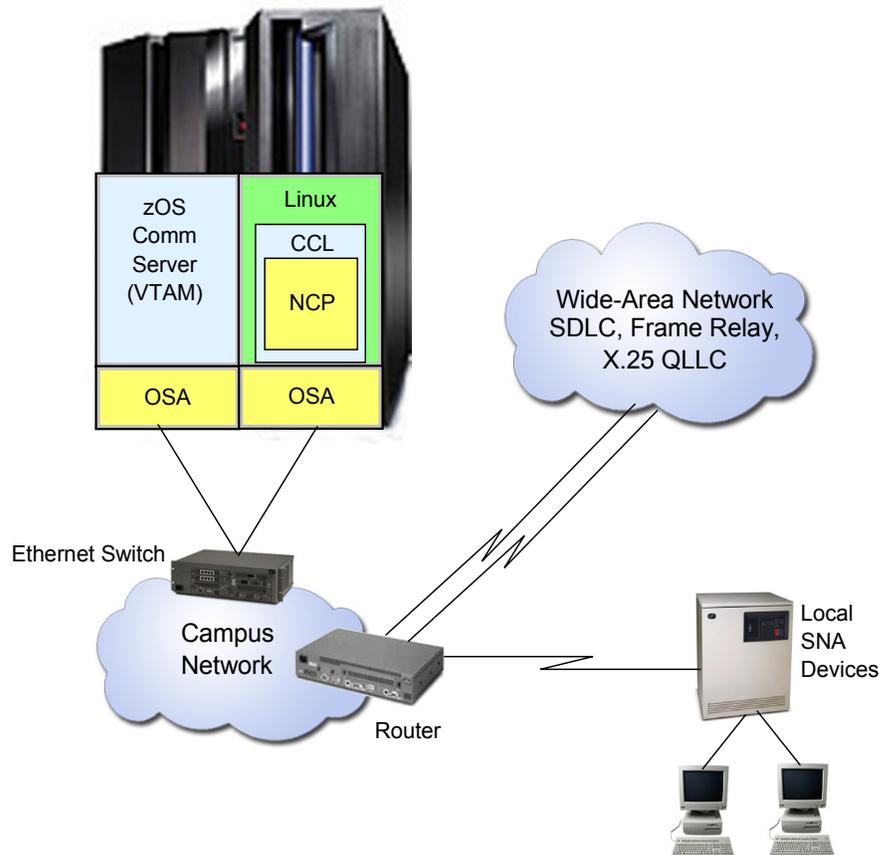


Figure 3-6 CCL network connectivity

### 3.8 HiperSockets

Mainframe HiperSockets™ is a technology that provides high-speed TCP/IP connectivity within a central processor complex. It eliminates the need for any physical cabling or external networking connection between servers running in different LPARs.

The communication is through the system memory of the processor, so servers are connected to form a “internal LAN.”

The HiperSockets implementation is based on the OSA-Express Queued Direct I/O (QDIO) protocol, hence HiperSockets is also called internal QDIO, or IQDIO.

The microcode emulates the link control layer of an OSA-Express QDIO interface.

Figure 3-7 on page 64 shows how to use HiperSockets; of particular note is that z/OS is not the only operating system running on a mainframe host that can take advantage of HiperSockets. Other operating systems include z/VM and Linux.

**Note:** z/VM is capable of functioning very similar to an LPAR. Instead of doing the work at the hardware level, z/VM creates separate virtual environments using software. It is highly efficient and it is sometimes used to run large numbers of separate and independent Linux hosts, all within a single LPAR.

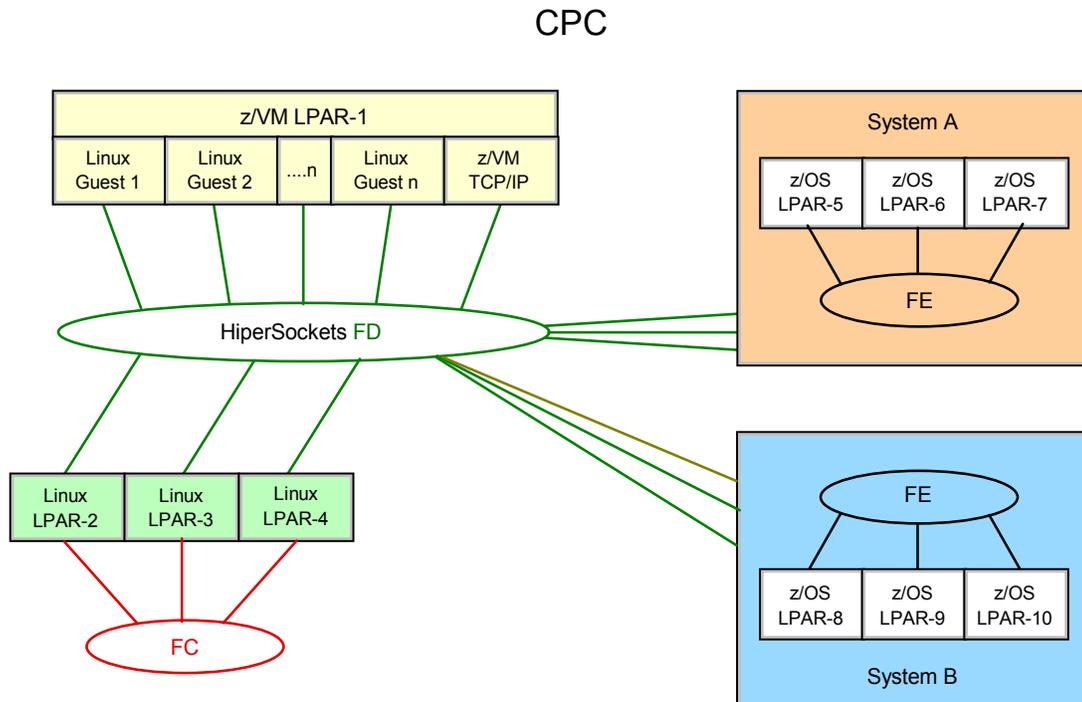


Figure 3-7 HiperSockets usage example

- ▶ HiperSockets with CHPID FC  
This HiperSockets channel path exclusively serves three Linux systems running in LPAR-2, LPAR-3, and LPAR- 4.
- ▶ HiperSockets with CHPID FD

Connected to this HiperSockets channel path are all servers in the mainframe CPC, which are:

- The three Linux servers in LPARs 2 to 4
- The multiple Linux servers running under z/VM in LPAR-1
- The z/VM TCP/IP stack running in LPAR-1
- All z/OS servers in sysplex A (LPARs 5 to 7) for non-sysplex traffic
- All z/OS servers in sysplex B (LPARs 8 to 10) for non-sysplex traffic

▶ HiperSockets with CHPID FE

This is the connection used by sysplex A (LPARs 5 to 7) to transport TCP/IP user-data traffic among the three sysplex LPARs.

▶ HiperSockets with CHPID FF

This is the connection used by sysplex B (LPARs 8 to 10) to transport TCP/IP user-data traffic among the three sysplex LPARs.

The CHPID type used for a HiperSockets connection is called IQD.

**Note:** So how fast is IQDIO? How about transferring data between z/OS and Linux using FTP? First, multiple FTP connections must be run in order to get close to utilizing the bandwidth of the interface. When transferring data between a z/OS LPAR and a Linux LPAR over IQDIO, 50 FTP connections produced a total throughput of 600 MBps.

The limitation here, however is not z/OS. In order to increase the throughput, the Linux limitation needs to be overcome. The test was expanded to FTP connections between a single z/OS LPAR and 4 separate Linux LPARs. With 120 total FTP connections (30 for each Linux LPAR), the IQDIO throughput was over 1200 MBps, or more than 1.2 GBps.

## 3.9 Making the connections

The connections to the central processor complex are made in a physical area of the processor frame called an *I/O cage*. Within the cage, OSA cards and memory modules (and other devices) are physically attached to the central processor complex. Parallel, FICON and ESCON connections are all made within the cage as well, using an adapter card.

Figure 3-8 on page 66 shows a photograph of an I/O cage. The two cards with numerous small black fiber channel connectors are ESCON cards. The next card to the right is an OSA card, with two RJ-45 connectors. The cards connected to the large black cables are inter-system coupling (ISC) cards for coupling links. To

the far right of the image is another ESCON card with several fiber optic cables connected to it.

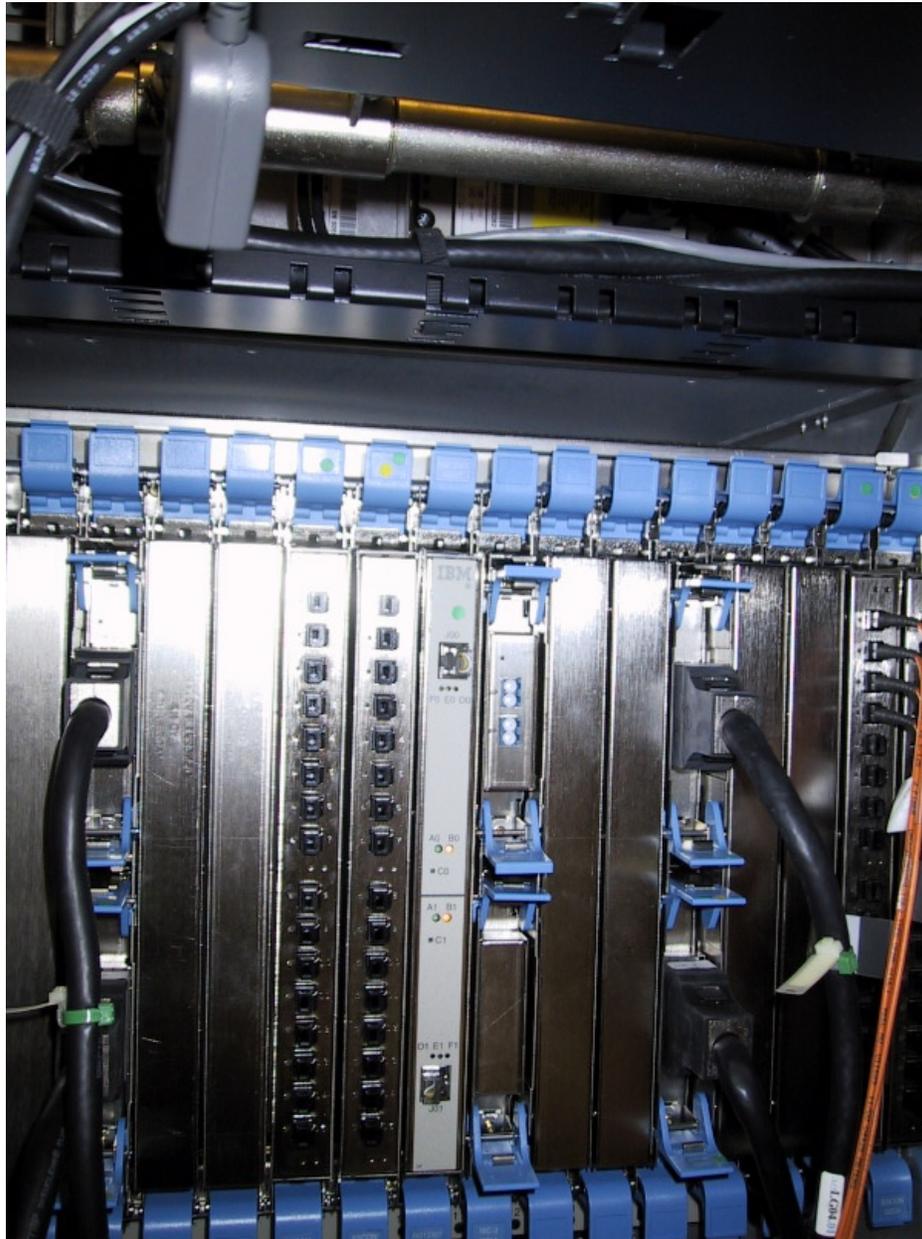


Figure 3-8 I/O cage

## 3.10 Summary

Network connections can be made in several different fashions. The mainframe originally relied upon the channel subsystem to offload I/O processing to channel programs. DASD is still accessed using ESCON channels, but for networking connectivity, OSA-Express cards offer better performance and availability.

The OSA-Express and OSA-Express2 cards provide redundancy capability, as well as throughput improvements when running in QDIO mode. QDIO mode allows direct access to central memory. QDIO mode can be emulated within a CPC by allowing memory to memory data transfer among LPARs running z/VM, Linux, or z/OS.

Key terms in this chapter		
central processor complex (CPC)	channel subsystem (CSS)	Communications Controller for Linux (CCL)
cross-system Coupling Facility (XCF)	direct access memory (DMA)	direct access storage device (DASD)
ESCON	FICON	HiperSockets
I/O cage	IQDIO	logical partition (LPAR)
Open Systems Adapter (OSA)	OSD	OSE
parallel channel	QDIO	VLAN





## Sample configuration

**Objective:** This chapter describes a real-world sample configuration that is used to demonstrate the various course concepts. Components of this configuration might be used in the hands-on exercises.

The objective of this chapter is to provide an introduction to the type of mainframe networking components and environment that many organizations deploy.

After completing this chapter, you will be able to:

- ▶ List the components of a typical mainframe networking infrastructure.
- ▶ Explain why organizations duplicate key components in the network.

## 4.1 Requirements for a mainframe network

As explained in Chapter 1, “Mainframes and networks” on page 3, businesses require their networks to be reliable, always available, and fast. They invest a great amount of time and money creating an IT infrastructure that supports these goals. The extent to which an organization implements a solution depends very much on availability and performance goals balanced against the cost of the solution.

The sample configuration in this chapter is designed to meet these goals, especially for availability.

## 4.2 The ZOS Company data center

Figure 4-1 on page 71 illustrates the concept of a medium-to-large z/OS data center. Processing is divided up physically by central processor complexes and logically by logical partitions, as described here:

- ▶ A *central processor complex* (CPC) is a physical collection of hardware that consists of main storage, one or more central processors, timers, and channels.
- ▶ A *logical partition* (LPAR) is a subset of a single physical system that contains resources (processors, memory, and input/output devices), and which operates as an independent system.

In reality, there would be more CPCs and LPARs. Most large organizations would duplicate this data center at another location as a backup for disaster recovery or site swap, in which processing is moved to this duplicate site. The disks would be also mirrored between sites.

Mainframe servers require minimal downtime, but sometimes microcode updates are required that might include resetting the server (this is called a *power-on reset*). There are also external influences that can cause an outage, such as changes to the data center infrastructure or router (connectivity) changes.



## Development LPAR

The Development/QA1 LPAR might also need a backup; this is insurance in case application programmers run tests on new applications that affect the LPAR. The development LPAR is used to develop new software.

Most large organizations have in-house programmers responsible for creating and maintaining applications that are specific to the organization's needs. These applications are created, tested, and maintained on the development LPAR.

## Test LPAR

A test LPAR is sometimes also referred to as a *system programming LPAR*. A test LPAR generally provides the basis for software delivery and early testing of changes and new functions. It is also where maintenance would be applied and tested prior to the fixes being implemented in production.

Ideally, the test LPAR should be as similar to the production LPAR as possible. This might then include a second systems programming LPAR on each CPC, providing an extra level of confidence when migrating changes through the system (though there is nothing like the real production acid test!).

The LPARs would also have network connections to each other by way of inter-CPC and intra-CPC hardware and software features.

## Isolating the production LPARs

Production LPARs are critical to maintaining an organization's viability.

Although Figure 4-1 on page 71 does not show it, the production LPARs are normally isolated logically (and sometimes physically) from the test and development LPARs. This is achieved by using a Parallel Sysplex, usually just called a sysplex. A *sysplex* is a clustering technique involving software and physical components. This technique helps with availability and workload balancing and protects environments from each other.

There would be a production sysplex for the production LPARs and a test or development sysplex for the remaining systems. Some organizations may have many sysplex systems. With the ability to define multiple independent sysplexes, even LPARs within a single CPC can be isolated logically by participating in separate sysplexes.

The main lesson to learn from Figure 4-1 on page 71 is that duplicate components are in place to allow for scheduled and unscheduled outages, and provide the availability that z/OS customers expect.

Table 4-1 provides a brief description of some of the components and terms included in Figure 4-1 on page 71.

Table 4-1 Components in the sample

Term	Description
CICS	Customer Information Control System. Provides transaction management functions and connectivity to application programs. Runs as an address space in z/OS.
CF	Coupling Facility. Enables sharing of data between multiple LPARs using high speed channels. Communicates LPAR status information.
CPC	Central processor complex. Physical collection of hardware and main storage and one or more central processors.
DB2	Database 2. Relational Database product used on most mainframe customer sites. Runs as an address space in z/OS.
OSA	Open Systems Adapter. High speed integrated cards used for network communication.
OSPF	Open Shortest Path First. Routing protocol used to communicate between router and mainframe TCP/IP OMPROUTE application.
TSO	Time Sharing Option. An element of z/OS that enables users to create an interactive session with the z/OS system. TSO provides a single-user logon capability and a basic command prompt interface to z/OS. Similar to a PC command prompt window.
VTAM	Virtual Telecommunications Access Method. The original SNA networking protocol for mainframes. Provides services to TCP/IP as well. Runs as an address space in z/OS.
TCP/IP	TCP/IP server address spaces.

Figure 4-1 on page 71 shows a TCP/IP stack and VTAM address space running under the z/OS operating system. It is normal to run only one instance of TCP/IP per LPAR, but some situations may dictate additional TCP/IP stacks on an LPAR (for example, where a business requires external partners to connect through a separate connection or network).

### 4.2.1 Key mainframe network availability aspects

Organizations buy mainframes for many reasons, but they generally fall into one or more of the following categories:

- ▶ Reliability, availability, serviceability (RAS)
- ▶ Security
- ▶ Scalability
- ▶ Continuing compatibility
- ▶ Evolving architecture

Table 4-2 looks at these categories from a z/OS networking perspective.

*Table 4-2 z/OS network availability aspects*

Category	Examples
RAS	<ul style="list-style-type: none"> <li>▶ High quality hardware and software components. Software put through rigorous compatibility testing.</li> <li>▶ OSA-E cards provide dynamic failure detection, takeover, and recovery.</li> <li>▶ VIPA (virtual IP address, which is not tied to any physical interface) provide movability and availability of IP addresses, independent of physical network adapters.</li> <li>▶ Many components, adapters, and disk units can be replaced, serviced non-disruptively.</li> </ul>
Security	<ul style="list-style-type: none"> <li>▶ Intrusion detection and control (through TCP/IP).</li> <li>▶ Monitoring and reporting features.</li> <li>▶ Network resource access control.</li> <li>▶ Cryptographic processors, network security protocol support.</li> <li>▶ External security control (through the System Authorization Facility).</li> </ul>
Scalability	<ul style="list-style-type: none"> <li>▶ Can handle very large customer loads and growth.</li> <li>▶ Dynamic non-disruptive expansion.</li> <li>▶ An example of an IBM internal test to show scalability: an environment with 64,000 concurrent Telnet 3270 sessions performing 6,242 transactions per second has been demonstrated successfully.</li> </ul>
Continuing compatibility	<ul style="list-style-type: none"> <li>▶ Compatibility of older applications, device types, and software with the newer releases of networking components is a key feature.</li> </ul>
Evolving architecture	<ul style="list-style-type: none"> <li>▶ The z/OS networking components are continually being developed.</li> <li>▶ Enterprise Extender was designed to assist with transporting SNA over IP networks.</li> <li>▶ Nondisruptive VIPA movement and distribution features assist with making the z/OS environment more robust and available.</li> </ul>

Key aspects that you as an mainframe network administrator should look for in terms of RAS might include:

<b>Component failure</b>	Each component should be analyzed for what would happen if this component failed, and it does not have a backup.
<b>Dual and diverse paths</b>	Is there more than one network path that provides an alternative route to a target, should a component fail?
<b>Performance</b>	Can the alternative component handle the load and performance on its own, should a failure occur on the primary component?  If load balancing, each component should have the capacity to takeover the load and performance, if required.
<b>Failure process</b>	How transparent would a component failure appear to a client of z/OS? The aim should be that any failure results in a non-disruptive dynamic change, that has minimal impact on the client. This is not always possible, but should be strived for.
<b>Security</b>	What are the client's requirements for securing network access?  This requirement might be within the z/OS security team's purview, but you might be required to implement TCP/IP or VTAM features to meet the security policy.

There are many more criteria that could be applied; for example, is the solution scalable and manageable, and will it meet the service level agreements (SLAs) agreed with the business?

**Reminder:** An SLA is a formal document between a service provider, such as the organization running the mainframe, and its customer, the recipient of the service. Customers of an SLA may be internal to the organization.

The extent to which organizations will go in order to ensure a high availability z/OS networking environment will vary.

Our sample network is designed with a focus on maximum availability. If a component fails, then another component should be able to continue in its place. There are hardware and software components that contribute to availability, as described here.

## Hardware availability

Figure 4-1 on page 71 contains the following components that contribute to hardware availability:

- Switches or routers** Two network switches, sometimes referred to as the *core network switches*, with multiple paths to each other and to the four OSA cards. Each switch would be able to handle the inbound and outbound data on its own to handle scheduled and unscheduled maintenance. The routers will be capable of running the OSPF routing protocol.
- OSA cards** Two OSA cards per CPC. These cards are defined and shared by both CPC LPARs. In most organizations there will be many more cards. LPAR1 and LPAR2 share OSA-E1 and OSA-E2 on CPCA. Each LPAR will have a unique IP interface address defined within the TCP/IP stack for OSA-E1 and OSA-E2 cards. The ability for the OSA cards to be shared provides flexibility should a card or switch fail. OSPF also plays a role here, as it can detect link state changes and switch to an alternate path, if required.
- CPC** Two mainframe CPCs to allow for any scheduled or unscheduled maintenance. The mainframe processors are very reliable, but some microcode updates require a power-on reset. Some changes, such as a new z/OS upgrade, by implication might involve swapping CPCs while some testing in isolation is carried out.
- Coupling Facility** The Coupling Facility is also normally duplicated, but a duplicate is not shown on Figure 4-1 on page 71. The CF is used for communication between z/OS LPARs in a sysplex.

## Software availability

Figure 4-1 on page 71 contains the following components that contribute to software availability:

- OSPF** The environment is running the OSPF routing protocol. Some organizations may use other routing protocols such as RIP. OSPF is a link state routing protocol; it runs on routers. Under z/OS, OSPF is implemented by running an application (started task) called OMPROUTE.

If OSA-E1 became unavailable on LPAR1, OSPF detects this and would send all LPAR1 traffic through OSA-E2 until OSA-E1 is restored.

OSPF within a mainframe data center environment is normally configured as a *stub area*. This is done to block external route and summary route information from being propagated down to z/OS, which only needs to know about adjacent routes.

#### **TCP/IP**

Each LPAR TCP/IP stack normally has allocated what is known as a *virtual IP address* (VIPA). A VIPA address is not tied to any physical interface, so it will never fail.

Applications and network hosts are normally configured to target the VIPA addresses.

VIPAs can be configured to move manually or dynamically. For example, LPAR2 could take over a VIPA address on LPAR1, and users would not lose their connections and would not be aware of the path change.

VIPAs are a key component in providing high availability solutions for organizations.

#### **VTAM**

There are many functions within a z/OS VTAM environment that can be configured to aid in the recovery or the takeover of resources in the event of an outage.

## **4.3 Summary**

This chapter introduced you to the typical z/OS networking environment that organizations would have.

Organizations run many of their mission-critical applications on the mainframe and system availability is a key factor in maintaining an organization's business. To meet this goal, organizations duplicate hardware and software components.

There is no single solution that fits all, but the general principal of hardware and software failover and dynamic takeover is very attractive to organizations.

The key points are:

- ▶ Most organizations will have two or more central processor complexes (CPCs) to allow for scheduled and unscheduled outages.
- ▶ Most organizations will have a geographically isolated site to allow for disaster recovery situations.
- ▶ OSA cards can be shared among LPARs on a CPC.

- ▶ TCP/IP VIPAs are not associated with a physical interface, and they assist in maintaining availability for applications and users.

Key terms in this chapter		
central processor complex (CPC)	Coupling Facility (CF)	Customer Information Control System (CICS)
DB2	logical partition (LPAR)	OMPROUTE
Open Systems Adapter (OSA)	reliability, availability, serviceability (RAS)	service level agreement (SLA)
stub area	switch	sysplex
virtual IP address (VIPA)	Virtual Telecommunications Access Method (VTAM)	



## Part 2

# TCP/IP implementation on the mainframe





## TCP/IP on z/OS

**Objective:** After completing this chapter, you will be able to:

- ▶ Explain the usage of the TCP/IP profile configuration file.
- ▶ Explain the basics of the FTP and the telnetd server configurations.
- ▶ Discuss the options for the resolver configuration.
- ▶ List some of the most common client applications.

## 5.1 The heart of the matter

The single entity that handles, and is required for, all IP-based communications in a z/OS environment is the TCP/IP daemon itself. The TCP/IP daemon implements the IP protocol stack and runs a huge number of IP applications to the same specifications as any other operating system might do. That's the beauty of TCP/IP.

For example, you can run FTP, telnet, SNMP, sendmail, NFS, HTTP servers, rlogin, SSH, BIND DNS (and the list goes on). All these are standard implementations, and many of these applications are ported from the industry standard source code. Any of these applications interact with the same applications running on essentially any other operating system.

Though TCP/IP sounds familiar, on z/OS there are differences.

First of all, TCP/IP does not really run as a daemon. That term was used in the first paragraph because from a casual viewpoint, the TCP/IP started task is the same as a daemon. It starts up and stays running for as long as the operator wants it to. It handles service requests of various kinds from within the operating system environment and from the network. So why, you may ask, can't these z/OS people just call it a daemon?

### 5.1.1 Control issues and the stack

Calling the TCP/IP started task a daemon is a little like calling a professional quality sound system a radio. It's a question of configurability and control. A daemon is started by a simple command line in a script file. There are few environment variables that can be specified.

By contrast, the TCP/IP started task is started as a program using Job Control Language (JCL). The capability of specifying parameters in JCL opens the door to a level of environmental control and configurability that cannot be matched in a daemon environment. Much of this chapter touches upon the JCL environment and how it can affect the way TCP/IP runs.

As an example, the z/OS environment can support more than one instance of a TCP/IP started task. The tasks are completely isolated from one another. In fact, you need to define links, either externally or internally, if you want the two TCP/IP tasks to have IP connectivity to each other. Without the JCL and the inherent z/OS underpinnings, such a scenario would be impossible.

The other reason why we call TCP/IP a started task is because the z/OS environment really does run daemons. In fact, we discuss the z/OS UNIX System Services telnet daemon in 5.4, "The telnet daemon" on page 94.

But we're not through with the TCP/IP started task just yet. TCP/IP is a *layered* protocol, and networking professionals want to keep that in mind. Consequently, the most commonly used term for the TCP/IP started task is the TCP/IP "stack." This is the terminology used in this chapter.

### 5.1.2 IPv4 versus IPv6

As of the time of writing, IPv6 continues to be the coming standard for IP addresses. However, it has not been widely implemented yet. Therefore, although this edition of the textbook uses examples and scenarios in an IPv4 context only, you should be aware that IPv6 is fully supported on z/OS.

## 5.2 The TCP/IP profile

Having extolled the versatility of JCL, it would be unfair not to provide a sample for starting the TCP/IP stack, so refer to Example 5-1.

*Example 5-1 Sample JCL for TCP/IP task*

---

```
//TCPIP PROC  
//TCPIP EXEC PGM=EZBTCPIP,  
//PROFILE DD DISP=SHR,DSN=SYS1.PARMLIB(PROFILE)  
//SYSTCPD DD DISP=SHR,DSN=SYS1.PARMLIB(TCPDATA)
```

---

This JCL sample is simplified and would not execute if submitted to the job entry subsystem (JES), but it contains the basic elements pertinent to TCP/IP. Most of the missing statements would be parameters unique to an organization's specific requirements.

The first statement identifies our JCL as a started task procedure. The second statement identifies the program to be executed, which in this case is EZBTCPIP.

**Tip:** The z/OS environment includes a convention of program (load module) and message prefixes that are generally standardized across each application environment. For the TCP/IP environment, the message and module prefix is always EZn. Hence many messages begin with EZA or EZB, and as we can see, the module that starts TCP/IP itself begins with EZB.

Because a z/OS console can be a very busy place, being able to recognize such a prefix can immediately provide a system operator with a context for a given message.

The third statement is a DD statement that assigns a pointer to the data set member PROFILE found within data set SYS1.PARMLIB. The TCP/IP started task automatically searches for a file (data set) allocated to (pointed to) by a DD statement (pointer handle) called PROFILE at startup.

There are other methods of allowing the TCP/IP program to search other locations for profile information. However, the first place that TCP/IP searches is the PROFILE DD statement. Other methods can be used if an organization has such a need.

The TCP/IP profile is read by TCP/IP when it is started. If a change needs to be made to the TCP/IP configuration after it has been started, TCP/IP can be made to re-read the profile dynamically (or read a new profile altogether).

### 5.2.1 Profile statements

The primary configuration point for the TCP/IP stack on z/OS is the TCP/IP profile data set. It defines operating characteristics of everything that is under the umbrella of the TCP/IP stack. Some of the more significant statements found in the TCP/IP profile are:

- ▶ IP link and address configuration (DEVICE, LINK and HOME statements)
- ▶ IP configuration (IPCONFIG statement)
- ▶ TCP configuration (TCPCONFIG statement)
- ▶ Static routing information (BEGINROUTES statement)
- ▶ Automated IP application monitoring (AUTOLOG)

It is impractical to discuss all statements found within the TCP/IP profile. Further, it is not even practical to discuss all parameters within each of the chosen statements. Keep in mind that even the statements discussed here have other parameters available that have been omitted in order to keep the discussion to a reasonable scope. All TCP/IP profile statements have default values that are used when they have not been explicitly coded.

#### Link configuration

TCP/IP supports more than a dozen different types of device attachments to the network. The most significant one is the OSA-Express adapter, discussed in Chapter 3, “Hardware connectivity on the mainframe” on page 45. There are two statements involved in configuring any adapter for TCP/IP on a z/OS host: a DEVICE statement and a LINK statement. In Example 5-2, there are three different adapters configured, and the first two of them are OSA-Express links.

### Example 5-2 Defining links

---

```
DEVICE OSAEDEV1 MPCIPA PRIROUTER  
LINK OSAELNK1 IPAQENET OSAEDEV1
```

```
DEVICE OSAEDEV2 MPCIPA PRIROUTER  
LINK OSAELNK2 IPAQENET OSAEDEV2
```

```
DEVICE VIPADEV1 VIRTUAL 1  
LINK VIPALNK1 VIRTUAL 1 VIPADEV1
```

```
HOME  
201.2.11.9 VIPALNK1  
201.2.11.1 OSAELNK1  
201.2.11.2 OSAELNK2
```

---

Because z/OS supports so many different hardware devices, the device and link statements are unique, depending upon the hardware they are defining. However, statements consistently follow a pattern in which the first parameter is the device or link name and the second parameter provides information with respect to that hardware being defined.

For example, the first DEVICE statement has been assigned OSAEDEV1 for a device name, and the device itself is going to be an MPCIPA type device. MPCIPA is the device type used by TCP/IP when the hardware is capable of QDIO-type communication. The subsequent LINK statement, assigned a name of OSAELNK1, further modifies the QDIO definition as IPAQENET. IPAQENET indicates that the link is a real OSA device. By “real”, this is to differentiate it from an IQDIO (remember, the I stands for internal) device, which would use a type of IPAIQDIO.

There are a few other options here that warrant special notice. The PRIROUTER parameter is necessary if this link is expected to handle packets destined for other networks or subnetworks. With PRIROUTER coded, the adapter not only accepts datagrams for IP addresses that are defined to the TCP/IP stack, but this parameter also causes the adapter to accept datagrams sent to it that are destined for any IP address. It is up to the TCPIP stack itself to then decide what to do with the packet (see “IP configuration” on page 87 for details about datagram forwarding behavior).

**Note:** The TCP/IP configuration of an OSA-Express device depends upon the prior appropriate hardware configuration through IOCP and VTAM.

The next statement of interest in this sample is the HOME statement. The HOME statement’s parameters consist of simple IP address and link name pairs. The HOME statement must include an IP address and link name pair for every hard-coded link active in the stack. For example, many z/OS hosts would have a

second OSA-Express card defined and its home address would be included in the HOME statement.

An OSA-Express device, as configured in this sample, downloads all IP addresses found in the HOME statement.

This brings us to the VIRTUAL DEVICE and LINK statements. TCP/IP on z/OS supports what is known as a virtual IP Address, or VIPA. Conceptually, VIPA is very simple: a VIPA address is functionally identical to any other IP address defined to the TCP/IP stack except that it is not dependent upon any physical networking hardware. A VIPA link is defined in software only; there is no physical hardware associated with it at all.

Why bother? Well, a VIPA address cannot suffer a hardware or LAN or link failure. It is always available. In our sample above, the user community would not be told to connect to the real links. Instead, clients would connect to 201.2.11.9, the VIPA address. If either of the two OSA links failed, the VIPA address would continue to be available using the adapter that still remained active.

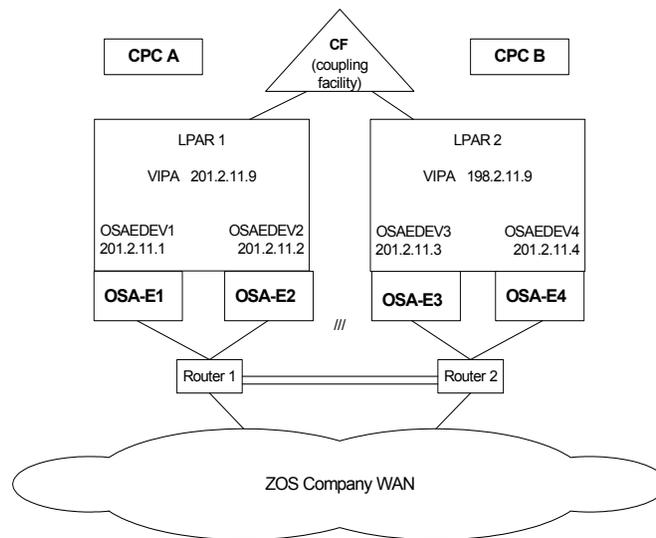


Figure 5-1 Examples of virtual IP addresses

In Figure 5-1 on page 86, there are two different examples of how a VIPA address can be placed within a network (routing) topology. In LPAR 1, the VIPA

link uses the same subnetwork as the OSA-Express links. From a network topology perspective, the VIPA address functions as an alias of either OSA card.

In LPAR 2, the VIPA address is on its own unique subnetwork. A router adjacent to this LPAR (in this sample, Router 2) would consider the VIPA address to be two hops away. The next hop for such a router would be to send the packet to one of the OSA-Express cards. The final hop, from the OSA-Express adapter's address (subnetwork) to the VIPA's address (or subnetwork), is completed internally in the z/OS IP stack.

The scenario used for LPAR 2 is the preferred scenario. If the VIPA address is placed in the same network as the physical adapters, then it must always remain part of a TCP/IP stack that has an adapter on that physical network. In other words, the VIPA becomes confined to remaining adjacent to the subnetwork of the physical network that it belongs to. When the VIPA is in its own unique subnet, it can be moved to any TCP/IP stack in the network (and dynamic routing takes care of locating it).

## IP configuration

The IPCONFIG statement group can be used to control characteristics related to the IP layer function of the TCP/IP stack. Some of these parameters are related to sysplex functioning and are discussed in Chapter 6, "TCP/IP in a sysplex" on page 103.

Of note in the IPCONFIG group are controls for datagram forwarding. *Forwarding* is the act of moving a datagram between two different networks or subnetworks. TCP/IP can be configured to not allow any datagrams to be forwarded. This prevents TCP/IP from inadvertently being used as a router. The option controlling this is the DATAGRAMFWD option.

As already noted, TCP/IP on z/OS is usually a multi-homed host. In other words, it has more than one IP address (link) associated with it. Quite often, it has at least one OSA-Express card connecting it to the network, with one or more specialized links connecting it to other hosts. If the links are redundant (that is, the routes to each link have equal costs), the IP layer can be configured to utilize all routes of equal cost. The option controlling this is the MULTIPATH option.

## TCP configuration

The configuration of TCP layer parameters is controlled with the TCPCONFIG parameter. The most significant parameters within this statement block are the parameters controlling the size of send and receive buffers. These parameters can have a significant impact on network performance, particularly when doing bulk data transfer. These parameters are:

## TCPMAXRCVBUFSIZE

An individual application running on z/OS can request to increase the default receive buffer size. This parameter limits the size that can be requested by any application. If there are a large number of TCP applications running on the z/OS host, this value could be significant.

By limiting the receive buffer size, TCPMAXRCVBUFSIZE can avoid an inbound flood of data arriving over multiple concurrent applications. In other words, the maximum buffer size is only significant as the number of applications using it become larger. The maximum is 512 KB.

**TCPRCVBUFSIZE** This is the default receive buffer size given to a TCP application on the z/OS host.

**TCPSENBFSIZE** This is the default for the size of the buffers used to hold outbound data prior to transmission. It can be increased up to 256 KB.

**Reminder:** A TCP buffer size correlates directly to a window size. When a session is established, each side sends out its receive buffer size. Each host uses the remote host's receive window size as an indication of the maximum amount of data that can be transmitted without an acknowledgement.

## Static routing information

The big advantage of static routing is the simplicity. A static route identifies a destination and the appropriate link to take to reach that destination. Static routing usually takes advantage of default routes: when the destination is not explicitly coded, send the packet to the default router and let that router figure out how to get the packet to its destination.

So static routing is easy—but not very resilient. Most installations would not use static routing. Instead, a dynamic routing protocol such as OSPF would be used (see 6.6, “Routing in a sysplex” on page 123 for details). Dynamic routing takes a little more effort to plan and set up, but once the planning stage is completed, the network effectively takes care of itself. It can also make more efficient use of network topology: instead of dumping everything to a default router, OSPF can take advantage of more intelligent network design.

For testing purposes and for smaller installations, static routing might be all that is required. There are also situations where an organization would use a combination of static and dynamic routes.

In z/OS, the statement block used to configure static routes is the BEGINROUTES statement. In Example 5-3, a sample of static routes that could be used for LPAR 1 are coded.

*Example 5-3 Static routes*

---

```
BEGINROUTES
ROUTE 201.2.11.0 255.255.255.0 =          OSAELNK1 MTU 1500
ROUTE 201.2.11.0 255.255.255.0 =          OSAELNK2 MTU 1500
ROUTE DEFAULT          201.2.11.100 OSAELNK1 MTU DEFAULTSIZE
ROUTE DEFAULT          201.2.11.100 OSAELNK2 MTU DEFAULTSIZE
ENDROUTES
```

---

Example 5-3 begins with two equivalent routes for reaching the directly-attached subnetwork 201.2.11.0. Following that are two default routes pointing to a router (not shown) with IP address 201.2.11.100.

Some redundancy is built into this scenario: either adapter can be used to reach the directly attached network and either adapter can be used to reach the default router. It sounds like static routes have some resiliency, right?

Well, the difficulty is in the fact that the second adapter is only used if TCP/IP on z/OS detects a failure with the link. For example, if OSAELNK1 were to be stopped manually, TCP/IP would immediately switch to using OSAELNK2.

However, if a problem occurred with one of the adapters and the problem was not detected by the OSA-Express card or by TCP/IP itself, then the adapter would continue to be used. It's better than nothing, of course, but a routing protocol like OSPF would automatically sense and make accommodations for the bad link.

The other difficulty with static routes occurs with VIPA when it is in a different subnetwork from the physical interface. How does an adjacent router (for example, 201.2.11.100 as shown in Example 5-3 on page 89) know that 198.2.11.9 (the VIPA in LPAR 2 from Figure 5-1 on page 86) is to be reached by sending a packet to 201.2.11.3 or 201.2.11.4? The answer is that the router needs to be told by coding a static route in its configuration. Again, in this context "OSPF is your friend."

## **Automated IP application monitoring**

All IP applications on z/OS require the TCP/IP stack to be running in order for IP communications to occur. It seems reasonable that when TCP/IP is started, the associated IP servers should be started at the same time. Better still, if TCP/IP can start these applications, why not have TCP/IP monitor them to make sure they continue to run correctly?

This can be accomplished by the AUTOLOG statement block (with assistance from a PORT statement block). The AUTOLOG statement contains a list of started task names that should be started and remain functional while TCP/IP itself is running.

But how does TCP/IP know whether an application is functional? It checks periodically to see if the application has an active listen on the appropriate port. If no listen is detected, TCP/IP stops the started task and then restarts it. By default, TCP/IP checks every 5 minutes for an active listen.

For example, if an organization wanted to have the FTP server automatically started by the TCP/IP task, and monitored every 5 minutes by the TCP/IP task, the definitions in Example 5-4 would accomplish this.

*Example 5-4 Autologging the FTP server*

---

```
AUTOLOG 5
  FTPD JOBNAME FTPD1
ENDAUTOLOG
PORT
  20 TCP OMVS NOAUTOLOG
  21 FTPD1
```

---

### **Generic servers and multiple TCP/IP instances**

Autologging has some limitations. For example, in an environment running more than one TCP/IP stack at the same time, an application like FTPD makes its services (listening port) available to all active TCP/IP stacks in the LPAR. In such a context, FTPD is referred to as a *generic server*. If FTPD is running with multiple TCP/IP stacks, then autologging could result in some confusion, as each TCP/IP stack attempts to stop and restart the FTPD server.

It is possible to override generic behavior and have a generic server associated with a specific TCP/IP stack only; *z/OS Communications Server IP Configuration Guide* describes which servers are generic and which ones always choose a specific TCP/IP stack for its services.

## **5.3 The FTP server**

FTP, like some other IP applications, is actually a z/OS UNIX System Services application. It can be started within an MVS™ environment, but it does not remain there very long. It immediately forks itself into the z/OS UNIX environment and tells the parent task to kill itself.

**A note on “MVS” versus “z/OS UNIX”:** The z/OS operating system has existed in one form or another for decades, and has been known by many other names since it was introduced in 1964 as “OS/360.” The most common of these older names is Multiple Virtual Storage or MVS. Even today, you will often hear z/OS system programmers use the term “MVS” to mean “z/OS.” It is not surprising that MVS is so deeply ingrained because the MVS era (1974-1992) was a major phase in z/OS history, and gave us innovations like multiple address spaces and virtual storage, among many others.

The z/OS operating system still provides the interfaces and system services of the original MVS operating system, but adds many other functions, including extensive support for UNIX interfaces and system services. z/OS essentially adds a UNIX environment through a system component called z/OS UNIX System Services.

For the purposes of this textbook, we still need to keep the MVS term around. We use the term “MVS services” in discussions of UNIX System Services whenever we need to refer to z/OS functions that operate aside from the UNIX environment in z/OS. The term “MVS” is important because it can be used to differentiate what would be “kernel” services in some other operating systems from “other” services (including UNIX ones). We use the term MVS services to include, for example, the z/OS system services provided by the z/OS base control program (BCP), and we refer to system error messages as MVS messages.

Because UNIX is an operating system on other platforms, you might wonder which of the two - MVS or z/OS UNIX System Services - is the real operating system. The answer is both, actually. MVS services and z/OS UNIX services are two sets of services available in z/OS and there are many others (such as TSO/E services and JES services). Most system functions of z/OS fall under the MVS heading, but the z/OS UNIX environment forms a significant subset.

Understanding this dual nature of z/OS is important to your ability to master the operating system. In this textbook, some applications are “MVS applications” in the sense that they do not use the z/OS UNIX environment. Other applications are “z/OS UNIX applications” because they must run as a UNIX application in the z/OS UNIX environment. Additionally, all socket communications are handled by z/OS UNIX services, not MVS services. Thus, you often see IP applications like FTP starting in the MVS environment and moving to the z/OS UNIX environment.

The FTPD task could very well be executed using `/usr/sbin/ftpd` and a few organizations probably do just that. However, if you recall “Automated IP application monitoring” on page 89, the FTP server can be autologged by the

TCP/IP started task if JCL is used. So, the FTP daemon is best started using JCL. An example can be found in Example 5-5.

*Example 5-5 FTPD started task JCL*

---

```
//FTPD  PROC MODULE='FTPD',PARMS=''
//FTPD  EXEC PGM=&MODULE,REGION=4096K,TIME=NOLIMIT,
//      PARM='/&PARMS'
//SYSFTPD DD DISP=SHR,DSN=SYS1.TCPPARMS(FTPSDATA)
```

---

Of interest to this discussion is the SYSFTPD DD statement. This DD points to the FTPD configuration file, usually referred to as the FTP.DATA data set. Like the TCP/IP started task itself, the FTP server searches other locations to locate the FTP.DATA configuration data set if SYSFTPD is not defined. However, in most cases using SYSFTPD is the preferred method.

What type of information is defined in the FTP.DATA data set? A hallmark of the z/OS operating system is configurability and the FTP server is no exception.

Here is a partial list and brief description of some of the more significant items that can be controlled:

- |                                |   |
|--------------------------------|---|
| <b>Banner page</b>             | Standard FTP server capabilities such as a banner page file are fully supported.  |
| <b>Anonymous configuration</b> | Several levels of anonymous logon control can be configured. When defined to ANONYMOUSLEVEL 3, an FTP user is given a unique root directory (a <b>chroot</b> command to change the root directory is executed by the FTP server) and further restrictions can be placed on other elements of the environment. |
| <b>Data set defaults</b>       | Data set attributes (block size, record format and much more) can be specified. An FTP client can override these attributes while in an FTP session.  |
| <b>Tracing and logging</b>     | Logging of users and detailed debug information can be activated. Output goes to syslogd (the z/OS UNIX log daemon). If syslogd is not running, then messages are automatically redirected to the z/OS system console log.  |
| <b>File system</b>             | When an FTP client logs on, the user can automatically be placed into either the z/OS UNIX hierarchical file system or else into the z/OS file system.  |
| <b>SSL/TLS</b>                 | A secure session can be optional or required when connecting to the FTP server.   |

## JES and DB2 environments

An FTP client can be used to interact with JES or DB2 on z/OS.

A small sample of an FTP.DATA file is shown in Example 5-6 on page 93. Any configuration statements not defined are assigned default values.

### *Example 5-6 Sample FTP.DATA statements*

---

```
BANNER /etc/ftp.banner
ANONYMOUSLEVEL 3
ANONYMOUSFILEACCESS HFS
FTPLOGGING TRUE
STARTDIRECTORY HFS
```

---

As mentioned, FTP server messages destined for syslogd are redirected to the z/OS system log when syslogd is not running. Tracing and logging of the FTP server can produce a significant amount of output. If the syslogd daemon is stopped, such FTP messages, while not harmful, could be an inconvenience.

**Reminder:** Syslogd on z/OS UNIX is the same daemon you find running on any other UNIX-style platform. It is a central repository to which applications on the host or network can direct messages. Messages can be classified by the type of message or the priority of a message.

## 5.3.1 Character sets - FTP server

From an end-user standpoint, FTPing into the z/OS UNIX HFS and FTPing into, say Linux, appears identical. The FTP subcommands are the same and the directory structures are displayed in the same fashion.

However, there is one fundamental difference between z/OS UNIX and all other UNIX style operating systems: z/OS UNIX does not use the ASCII (American National Standard Code for Information Interchange) character set. Instead, z/OS UNIX is implemented using the EBCDIC (Extended Binary Coded Decimal Interchange) character set.

What are the implications? The conversion in most contexts is transparent: messages and commands are automatically translated going in either direction. An end user of an ASCII FTP client would see no difference of appearance between a z/OS FTP server and any other FTP server.

For file transfers, the translation is automatic when using ASCII mode (also called *character mode* or *text mode*). If an FTP client sends an ASCII encoded file to z/OS (either to the HFS or the MVS file system), the file is translated into

EBCDIC prior to being written to disk. If an FTP client retrieves such a file, it is automatically translated back into ASCII after being read from disk but prior to transmission. An FTP client would never know the other endpoint was storing the file as EBCDIC.

With a binary transfer, no translation occurs. Consequently, if an FTP client sends an ASCII file in binary mode to the z/OS host, this file does not display correctly when viewed on the z/OS host. If the FTP client retrieves an EBCDIC file using a binary transfer type, then the file does not display correctly when it arrives at the host of the ASCII FTP client.

As z/OS continues to evolve, some limited localized support of native ASCII is available. One example is referred to as *file tagging*, which can be used to identify a file's character set.

## 5.4 The telnet daemon

There are two telnet servers available in the z/OS operating environment. The TN3270 server supports line mode telnet, but it is seldom used for just that. Instead, it is primarily used to support the TN3270 Enhanced protocol; refer to Chapter 10, “TN3270 Enhanced” on page 241, for more details.

The other telnet server is a line mode server only. This chapter deals with the telnet line mode server, referred to as the z/OS UNIX Telnet server, or `otelnetd`. The `otelnetd` server is a functional equivalent to the `telnetd` daemons found on other UNIX style servers. Since it is a z/OS UNIX daemon, it is run within the z/OS UNIX environment and provides the user with a shell only (no access to the MVS file system is possible). All of the functionality is there that would be expected when connecting to a UNIX `telnetd` server: the vi editor, environment variables, profile processing, and more.

Additionally, like `telnetd` on other UNIX platforms, `otelnetd` requires `inetd` to handle its listen.

**Reminder:** What is `inetd`, anyway? The `inetd` server is a generic listening application. It is configured to listen on a specific port on behalf of another application, like a `telnetd` server. When a connect request arrives at that port, the `inetd` server starts an instance of the application and gives the connection to the newly started application. On z/OS, `inetd` also handles listens for industry standard implementations of `rlogin`, `rshd` and `rexecd`.

The bad news is that `inetd` does not lend itself readily to the AUTOLOG capability of the TCP/IP started task. The good news is that `otelnetd` is not typically used for mission-critical type communication anyway.

If you are familiar with configuring `inetd` for `telnet` on a UNIX style server, then you can certainly be comfortable configuring it on z/OS. A sample of `/etc/inetd.conf` is shown in Example 5-7.

*Example 5-7 inetd.conf for otelnetd*

---

```
#=====
# service | sock | prot | wait/ | user | server | server program
# name    | type |      | nowait|      | program | arguments
#=====
otelnet  stream tcp      nowait OMVSKERN /usr/sbin/otelnetd otelnetd -m
```

---

When a connection is made to port 23, `inetd` executes `/usr/sbin/otelnetd` and passes the parameter `-m` to it. The `-m` parameter is included for efficiency: it reduces the number of address spaces (processes) required to run the `otelnetd` session.

But wait a minute! How did port 23 suddenly get dropped into the equation? The `otelnetd` server on z/OS functions exactly the same as `telnetd` on any other UNIX platform. It requires an entry in `/etc/services` to inform it of which port number to listen on. In this case, `/etc/services` would require a line such as `otelnet 23/tcp`.

**Reminder:** The `/etc/services` file is used in a UNIX environment by many IP-based applications. When a UNIX application is started, it must determine what port number it should be listening on, and it checks `/etc/services`.

The `otelnetd` daemon supports the standard line mode terminal definitions such as vt terminals, Wyse terminals, PC850, and many others.

### 5.4.1 Character sets - telnet daemon

The EBCDIC-to-ASCII conversion with `otelnetd` is completely transparent. All characters sent to a client emulator are transmitted in ASCII format.

## 5.5 A good resolver is hard to find

For readers familiar with TCP/IP implementation on UNIX-style platforms, you might have experience with a file called `/etc/resolv.conf`. The `resolv.conf` file is referred to as a *resolver configuration file* or the `TCPIP.DATA` file.

So why are there two names? It harkens back to our discussion of MVS and its UNIX subset; it all depends on whether the application is a z/OS UNIX application or a native MVS application.

First, however, we need to describe how a resolver configuration file is used. The TCP/IP profile data set defines the operating characteristics of the TCP/IP stack. However, that is not quite enough. There are a few operating characteristics associated with IP applications that must also be defined. The resolver configuration file defines the operating characteristics of IP applications. A sample resolver configuration file can be seen in Example 5-8.

*Example 5-8 Sample resolver configuration file*

---

```
DOMAIN XYZ.COM
HOSTNAME MAINFRAME
NAMESERVER 200.1.1.1 200.1.1.2
TCPIPJOBNAME TCPIP
```

---

Most of these statements should be recognizable to those familiar with IP on UNIX-style systems.

<b>DOMAIN</b>	This statement identifies the domain name system (DNS) domain origin for this host.
<b>HOSTNAME</b>	The host name (highest qualifier) of the fully qualified host name for this host. When concatenated together, the host name in Example 5-8 is MAINFRAME.XYZ.COM. (The domain name system is not case sensitive, by the way.)
<b>NAMESERVER</b>	If an IP application needs to resolve a host name to an IP address (or the opposite), it uses one of these IP addresses to contact a name server. Often, on z/OS, the statement NSINTERADDR is used instead of NAMESERVER.
<b>TCPIPJOBNAME</b>	This statement identifies the started task name (job name) of the TCP/IP stack that an application can be associated with. It might not be used but should always be coded!

**Note:** What is meant by stating that the TCPIPJOBNAME statement “might be used?” In an MVS environment, the TCPIPJOBNAME statement is used only if the application has not explicitly overridden it (applications, except for Pascal applications, can explicitly request a particular stack name).

In a z/OS UNIX environment, TCPIPJOBNAME is never used. An application must explicitly choose a stack name or else it is associated with all active TCP/IP stacks. In other words, it is a generic server.

## 5.5.1 Searching for resolver configuration information

So how does an application, like our FTP server or the otelnetd daemon, find its resolver configuration information? This topic covers some simple options that work well, assuming that the system uses a single TCP/IP stack environment (only one active TCP/IP stack within the LPAR). Where the information comes from depends on:

- ▶ Settings in the resolver address space itself. The very first location checked for all applications is the GLOBALTCPIPDATA. The GLOBALTCPIPDATA data set is discussed in “Resolver address space.”
- ▶ The type of application (MVS or z/OS UNIX). This is discussed in “Type of application” on page 98.
- ▶ The search order for dynamically locating files. This is discussed in “The search order” on page 99.

### Resolver address space

The statements found in a resolver configuration file can be globally overridden at the z/OS level by the resolver address space.

When an application needs to access services identified within the resolver configuration file, it is accomplished using the resolver started task (address space). The resolver address space is normally started when z/OS is started. The significance of this resolver address space is that the address space itself can be configured with resolver configuration file statements.

These statements can be used to provide resolver configuration file settings for both z/OS UNIX applications and MVS applications. A sample of the JCL for starting the resolver configuration file is shown in Example 5-9. Again, this JCL is only a portion of the JCL required.

*Example 5-9 Sample JCL for starting resolver address space*

---

```
//RESOLVER PROC PARMS='CTRACE(CTIRES00) '  
//EZBREINI EXEC PGM=EZBREINI,REGION=0M,TIME=1440,PARM=&PARMS
```

```
//SETUP DD PATH='/etc/setup.resolver',PATHOPTS=(ORDONLY)
```

---

The statement of interest in Example 5-9 is the SETUP DD statement. This DD points to a file containing statements controlling the behavior of resolver configuration file searches. There are several statements available within a resolver configuration SETUP file.

Two more important statements are:

#### **GLOBALTCPIPDATA**

This statement is used to identify a specific resolver configuration file that contains the resolver configuration statements (NAMESERVER, HOSTNAME, and so on) that are to be applied globally to all IP applications.

#### **DEFAULTTCPIPDATA**

This statement is used to define a default resolver configuration file that is used as a last resort.

**Note:** Hopefully, that SETUP DD statement in Example 5-9 has not confused you. A DD statement can point to a z/OS UNIX file in the HFS as well as to a z/OS data set.

In this example, there is no particular reason why an HFS file was chosen; it could just as easily have been a z/OS data set or a data set member.

### **Type of application**

As alluded to previously, the location of the resolver configuration files is complicated by the dual nature of applications on a z/OS host. Where an application searches for a resolver configuration file depends on whether the application is an MVS application or a z/OS UNIX application.

For z/OS UNIX applications, the resolver configuration parameters can be placed in the obvious location: /etc/resolv.conf. However, this is not the very first place that a z/OS UNIX application searches. It is possible to identify the resolver configuration file in the environment variable RESOLVER\_CONFIG, but this variable would need to be defined to every z/OS UNIX IP application. Obviously, /etc/resolv.conf is simpler to use, yet some organizations may prefer to use or require the RESOLVER\_CONFIG environment variable.

**Note:** There is a benefit to having RESOLVER\_CONFIG ahead of /etc/resolv.conf in the search order.

For example, imagine if you wanted to run a z/OS UNIX application that used a different name server other than what is coded in /etc/resolv.conf. This can readily be accomplished by specifying the RESOLVER\_CONFIG environment variable and pointing it to a copy of /etc/resolv.conf containing a different value for NAMESERVER.

## The search order

To summarize, GLOBALTCPDATA is used for both z/OS UNIX and MVS IP applications. Then, z/OS UNIX applications look for RESOLVER\_CONFIG followed by /etc/resolver.conf. At this point, the remaining locations where an application might search for resolver configuration information is the same for both z/OS UNIX and MVS IP applications.

Here they are in the order in which they are searched for:

1. //SYSTCPD DD card. The data set allocated to the DDname SYSTCPD is used. In the z/OS UNIX environment, a child process does not have access to the SYSTCPD DD. This is because the SYSTCPD allocation is not inherited from the parent process over the fork() or exec function calls.
2. userid.TCPIP.DATA. “userid” is the user ID that is associated with the current security environment (address space or task/thread). An MVS environment application could theoretically run without an associated user ID. If so, the job name would be used for this data set instead.
3. SYS1.TCPPARMS(TCPDATA)
4. DEFAULTTCPDATA. If defined, the resolver DEFAULTTCPDATA setup statement value is used; see “Resolver address space” on page 97.
5. TCPIP.TCPIP.DATA

Only the first file located is used. For example, if the first file found is SYS1.TCPPARMS(TCPDATA), then that is the only file used, even if DEFAULTTCPDATA and TCPIP.TCPIP.DATA exist.

Not all statements within resolver configuration files are treated equally. Some statements not globally defined (in GLOBALTCPDATA) can be located dynamically in files later in the search order. For example, if TCPIPJOBNAME is not located in the global file, it can be read in from the first of the subsequent files in the search order.

This makes sense because if more than one TCP/IP task is running, defining TCPIPJOBNAME in GLOBALTCPDATA would only allow one TCP/IP task to

ever be accessed. By using something like RESOLVER\_CONFIG or SYSTCPD, different job names could be specified for different environments or applications.

But not all of the statements get this second chance. Some statements, if not defined explicitly in a global file, are forced into default values. For example, the NAMESERVER statement, if not present in the GLOBALTCPIPDATA, defaults to a blank, disabling all name server function.

When configuration time comes, each statement that is needed for an application should be reviewed to determine where it would best be located.

## 5.5.2 The multi-stack environment

Why does something as simple as searching for an application's IP environment information need to be so complex? The answer lies only partially in the dual nature of the z/OS (MVS and UNIX) operating system.

The other reason is that some organizations have a need to run more than one TCP/IP stack within a single LPAR. In such a context, there are some applications (servers) that should be associated with one TCP/IP stack and not the other TCP/IP stack. There are many different ways that this can be controlled, and some of it depends upon the nature of the application itself. But for some applications it is controlled by the TCPIPJOBNAME statement found in the resolver configuration file.

This implies that an organization would need to have more than one resolver configuration file: one for application A, which uses TCP/IP stack X and another resolver configuration file for application B, which uses stack Y.

The other reason for the complexity—and this applies to essentially any aspect of z/OS—is the configurability requirements. The z/OS operating system is designed for the largest organizations in the world. Such organizations are often also less flexible, more process-oriented, and security-aware environments. Consequently, z/OS must fit into the organization's requirements, and not the other way around.

What does this really mean? Well, if XYZ Corporation wants to use a certain naming convention for certain data sets because thousands of users security profiles are written with this in mind, then z/OS TCP/IP had better be able to conform.

## 5.6 TCP/IP clients

So far, we have mentioned only two server applications: FTP and telnet. We noted that z/OS supports all the well-known servers. In addition z/OS supports all of the well-known IP client applications.

Some of these applications are:

**ftp** The FTP client can be run from either a TSO environment or from the z/OS UNIX environment. The FTP client is heavily used on z/OS because it runs very well as a batch job. Large file transfers are scheduled for running on weekends or in the evenings and the JCL is submitted, and often evaluated for success, automatically.

**telnet** The line mode telnet client is a TSO application only. This application does not get a tremendous amount of use because z/OS is not a platform that lends itself to being a client (using z/OS to run your telnet client is overkill).

**ping** No IP implementation is complete without the ability to perform rudimentary connectivity tests. The `ping` command can be run from either the TSO or z/OS UNIX environments.

### **tracerte or traceroute**

On the MVS side of z/OS, there is a limit of eight characters for the length of a command. So, `traceroute` had to be shortened to `tracerte` when executed from a TSO environment. However, within the z/OS UNIX environment, the command is `traceroute` as would be expected. The `traceroute` command performs the UDP expired datagram method of testing the reachability of every hop in a network path.

**snmp** If an end user wants to perform rudimentary Simple Network Management Protocol queries, the z/OS UNIX `snmp` command can be used. It can also function as a trap (alert message) destination, but on z/OS, there are far better SNMP management environments available.

**netstat** The `netstat` command is probably the most essential IP command used by network administrators. Technically, it is not a client at all, but it is such a well-known and well-used tool that it is included in this list. It runs as a z/OS UNIX command, a TSO command, and it can also be issued as an z/OS system console command.

### 5.6.1 Character sets

Although the mainframe uses the EBCDIC character set, this is not an issue, as all TCP/IP clients do the translation from ASCII to EBCDIC and from EBCDIC

to ASCII automatically and transparently to both the end user and the remote application with which it communicates. Of course, some of these applications, like ping, traceroute and netstat, do not require any translation at all.

## 5.7 Summary

The TCP/IP started task is the engine that drives all IP-based activity on z/OS. The TCP/IP profile data set controls the configuration of the TCP/IP environment.

The FTP server implements the FTP standard and can communicate with any FTP clients on the network. The telnet server implements a standard line mode telnet daemon.

Even though z/OS is an EBCDIC host, communication with ASCII-based IP applications is seamless.

IP applications running on z/OS use a resolver configuration file for environmental values. Locating a resolver configuration file is somewhat complicated by the dual operating system nature of z/OS (UNIX and MVS).

TCP/IP on z/OS supports all of the well-known server and client applications.

Key terms in this chapter		
ASCII	Datagram forwarding	EBCDIC
FTPD	inetd	netstat
otelnetd	Receive buffer size	Send buffer size
resolver	ftp	ping
TCP/IP profile	traceroute	window size
snmp	telnet	



## TCP/IP in a sysplex

**Objectives:** After completing this chapter, you will be able to:

- ▶ Describe the key features of a z/OS sysplex.
- ▶ Explain how dynamic VIPA functions in a sysplex.
- ▶ Explain how dynamic XCF functions in a sysplex.
- ▶ Explain how sysplex distributor functions in a sysplex.
- ▶ Describe how the sysplex performs problem detection and recovery.
- ▶ Define the routing needs of a sysplex.

## 6.1 Computer cluster

A *cluster* of computers refers to a group of interconnected computers that are working together as one unit. It is a relatively simple concept and it has been around, arguably, for a very long time. Within a cluster, individual computers can be coupled either loosely or tightly.

- ▶ A *loosely coupled* cluster consists of computers that are running with a minimum of communication and cooperation among them. This results in efficient usage of each individual computer, but limits the amount of coordination and sharing of workloads.
- ▶ A *tightly-coupled* cluster consists of a group of computers that are cooperating to a great degree among each other, coordinating and sharing workload, and communicating status details on a continuous basis.

## 6.2 The z/OS sysplex

A *sysplex* refers to a tightly-coupled cluster of independent instances of the z/OS operating system. It is beyond the scope of this text to go into details on how the sysplex functions. Instead, this section will help you understand the aspects of a sysplex as it applies to TCP/IP.

A sysplex can be either *basic* or *parallel*. A basic sysplex can communicate using channel to channel (CTC) connections between LPARs. Parallel Sysplex uses something called a Coupling Facility (CF).

**Note:** The Parallel Sysplex Coupling Facility can run either as a separate LPAR or within a dedicated hardware device. It is capable of managing data structures on behalf of applications requiring inter-LPAR communication.

Logically speaking, a sysplex is a group of distinct instances of the z/OS operating system. A distinct instance is often called an *image*. The z/OS images could be running in separate CPCs, or they could be running in separate LPARs within a single CPC, or it could be combination of both: a few LPARs in one CPC and a few LPARs in another CPC.

What makes a group of such z/OS images into a sysplex is the inter-image communication. This inter-image communication is normally handled through the cross-system Coupling Facility, or XCF.

XCF communications function in either a basic sysplex or in a Parallel Sysplex. If a Coupling Facility exists, a choice can be made as to whether XCF uses CTCs or the Coupling Facility to communicate.

## 6.2.1 Cross-system Coupling Facility (XCF)

At the heart of the sysplex is the XCF. As its name suggests, XCF handles communication between logical partitions (LPARs) or CPCs. Communication between these LPARs is effectively (from a TCP/IP perspective) instantaneous.

Information such as workload, status, and data transmission occurs through the Coupling Facility. The information sharing is constant and continuous, allowing the independent z/OS images to know detailed information about the current status of all images within the sysplex.

TCP/IP takes advantage of the communication capabilities of the XCF in a sysplex in three different ways:

1. It maintains awareness of the status (health) of a TCP/IP instance within the sysplex.
2. It determines workload levels within each LPAR in the sysplex through Workload Manager (WLM).
3. It can send IP traffic among the LPARs.

In order for TCP/IP to be able to utilize the services of the Coupling Facility, VTAM must be configured to support XCF connections. This is normally accomplished by starting VTAM with XCFINIT=YES specified in ATCSTRxx. VTAM is covered in detail in Chapter 8, "SNA Advanced Peer-to-Peer Networking (APPN)" on page 165

As an implementation note, the Coupling Facility, if used, represents a single point of failure within a sysplex. Hence, an organization always implements a backup Coupling Facility which is ready in the event of a failure.

## 6.2.2 Workload Manager (WLM)

Now let's take a look at z/OS Workload Manager (WLM). WLM might be described as a *performance expectation facility*. It can be used to define performance goals (for example, response time) for different applications and different types of work.

In addition, WLM can be used to provide information about the overall workload levels on each image within a sysplex. For more information on sysplex and WLM, see *Introduction to the New Mainframe: z/OS Basics*.

By using WLM within a sysplex, a TCP/IP stack on z/OS can be configured to direct traffic to the LPAR with the lightest workload. The mechanism is relatively simple from the perspective of the TCP/IP application: TCP/IP asks WLM to provide information on workload levels within each LPAR in the sysplex. WLM

responds with a list containing the LPAR name (system name) and a number for each LPAR. Each number indicates the number of connections that should be sent to each LPAR for this time interval.

In the event that a sysplex becomes fully utilized, WLM can be configured to prioritize workload. In turn, TCP/IP distributes connections to the host with the lower priority workload. Or, if so configured, WLM can also distribute work based upon specific service class goals of the target application's address space. This is referred to as *server-specific WLM recommendation*.

## 6.3 Dynamic virtual addressing

At this point, the sysplex has been identified as a tightly-coupled cluster of computers with some workload balancing capabilities added for good measure. But how does the IP network take advantage of all this technology? The first part of the answer is the dynamic virtual IP address, or DVIPA.

DVIPA is part of the evolution of the VIPA feature we discuss in Chapter 5, "TCP/IP on z/OS" on page 81. The VIPA used in that chapter was static. It is defined through a DEVICE and LINK statement pair and remains unchanged unless explicitly removed by changing the active configuration statements.

By contrast, a dynamic VIPA would normally be activated in one of two different ways:

- ▶ An application explicitly issuing a bind() function call to the IP address. This is called *unique application-instance DVIPA*.
- ▶ A TCP/IP stack dynamically activating the address. This is called *multiple application-instance DVIPA*.

In order for TCP/IP to communicate DVIPA status among LPARs, TCP/IP uses an XCF group called EZBTCPCS.

### 6.3.1 Unique application-instance DVIPA

When an IP application intends to listen, or if an application intends to connect using a specific port or IP address, it must issue a bind() function call. The bind() function call can perform two services:

- ▶ Associate a socket with a specific IP address
- ▶ Associate a socket with a specific port number

**Reminder:** A socket is defined as a four-tuple consisting of:

- ▶ A local IP address and local port number
- ▶ A remote IP address and remote port number

For our purposes, we focus only on how a bind() can be used by an application to associate a socket with a specific IP address. An outline of this process is illustrated in Figure 6-1.

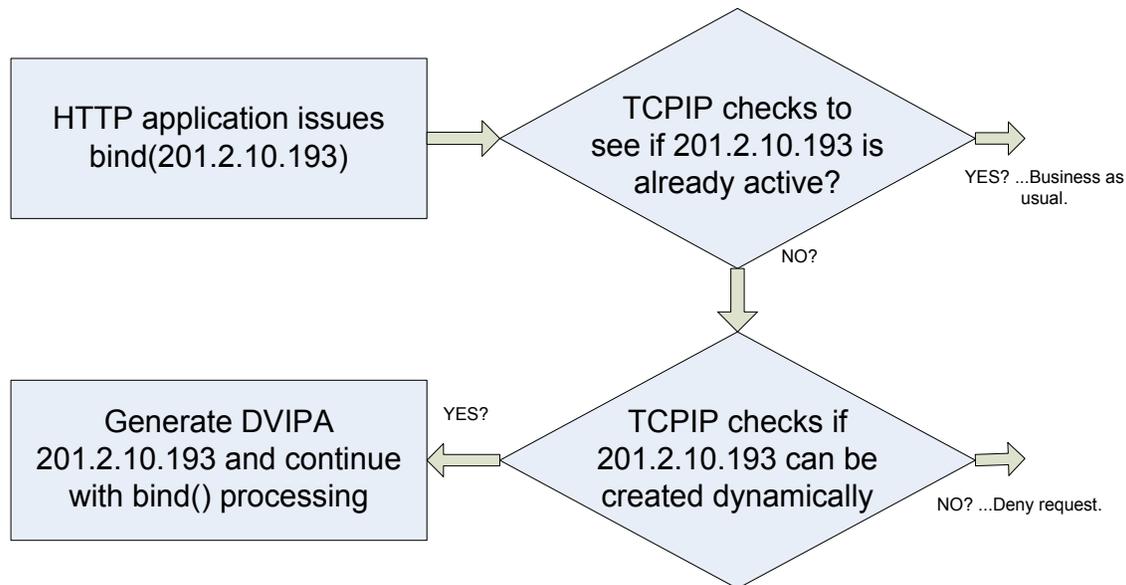


Figure 6-1 Single application instance DVIPA creation

The process begins with an application requesting the usage of a specific IP address. For example, one of the HTTP servers that runs on z/OS has a parameter in its `/etc/httpd.conf` file to force the server to use a specific IP address. When the HTTP server requests this IP address, the TCP/IP stack first checks to see if the address is already active.

This address might already be active because of one of the following reasons:

- ▶ It has been defined by a static definition.
- ▶ It has been dynamically created by another application.
- ▶ It has been dynamically created by the TCP/IP stack (multiple application-instance DVIPA, see “Multiple application-instance DVIPA” on page 110).

If the address is already active, the bind is processed with no exceptional processing required.

If the address does not exist, the TCP/IP stack next checks to see if this address is eligible for dynamic activation. If not, the bind is denied.

If the IP address is eligible for activation, then the TCP/IP stack dynamically generates the address. The application successfully binds to the IP address as though the address had always been there.

**Note:** Some applications issue bind() function calls without specifying any IP address at all. If desired, TCP/IP can be configured using the PORT statement with the BIND ip\_address keyword to override such a bind. TCP/IP substitutes the specified ip\_address and the application instance DVIPA processing as described in Figure 6-1 applies.

## Enabling DVIPA on a TCP/IP stack

So what is it that makes an IP address eligible for dynamic activation? As described in Chapter 5, “TCP/IP on z/OS” on page 81, the creation of a static VIPA address requires a DEVICE, LINK, and HOME statement. This is not the case for a DVIPA. To illustrate, let’s work our way through a set of statements for a simple DVIPA scenario, as shown in Example 6-1.

*Example 6-1 TCP/IP profile statements for DVIPA*

---

```
VIPADYNAMIC
  VIPARANGE DEFINE MOVEABLE NONDISRUPTIVE 255.255.255.192 201.2.10.192
ENDVIPADYNAMIC
```

---

All of the DVIPA-related statements are encompassed by a beginning (VIPADYNAMIC) and ending (ENDVIPADYNAMIC) statement. The VIPARANGE statement is exactly what its name implies: it defines a range of IP addresses that are eligible to be dynamically activated. In Example 6-1, the range is for IP addresses within the 201.2.10.192 network ID, using a mask of 255.255.255.192.

Tying this back to our HTTP server bind request from Figure 6-1 on page 107, you can see that the HTTP server has been configured to request 201.2.10.193. Since this IP address is within our VIPARANGE defined subnetwork, the bind() request results in a new IP address for a DVIPA being activated.

## Moving the DVIPA

The VIPARANGE statement has further options to modify its behavior in a context where other applications within a sysplex are also issuing bind() function calls. In our above example, how should this IP address behave if another

TCP/IP stack in the same sysplex (with the same VIPARANGE statement in its TCP/IP profile) issues a bind(201.2.10.193)?

By default, a VIPARANGE statement specifies that if another TCP/IP stack requests an activation of the same address, the movement is done in a non-disruptive fashion. This means that existing sockets using this IP address would be maintained. However, any new connection attempts to this address is directed to the TCP/IP stack that has most recently activated the VIPA address. After all sockets using the original TCP/IP stack have been closed, the VIPA address no longer exists on the original image.

**Note:** As the name implies, a unique application-instance DVIPA functions best when the DVIPA is activated by a single instance of an application.

The circle can be completed if the original application were to be restarted, causing it to issue a new bind to 201.2.10.193. The DVIPA would be moved back to the original stack in the same non-disruptive fashion.

Figure 6-2 shows a system able to continue HTTP processing if an HTTP server fails or needs servicing or maintenance. In LPAR 1, the HTTP server is bound to a dynamic VIPA address. In the event of a failure or a need to do service or maintenance on LPAR 1, there is an identically-configured HTTP server waiting to be started on LPAR 2.

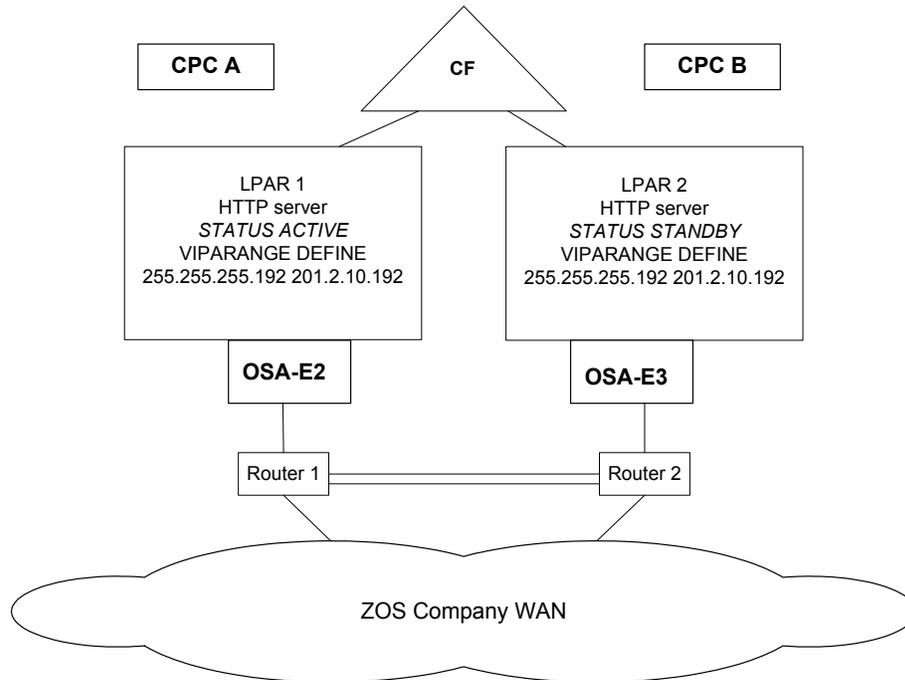


Figure 6-2 Unique application-instance DVIPA scenario

There are many automation packages, such as Tivoli® NetView® for z/OS and System Automation for z/OS, that can start another server automatically when a current server has failed or ended. So the system can automatically initialize a new HTTP server on LPAR 2 if the HTTP server on LPAR 1 ends. In turn, the act of starting the HTTP server on LPAR 2 is enough to initiate a non-disruptive movement of the IP address.

### 6.3.2 Multiple application-instance DVIPA

Multiple application-instance DVIPA is conceptually the same as unique application instance DVIPA. The difference is in the layer at which the DVIPA is associated. With multiple application-instance DVIPA, the VIPA address is activated by and associated with the entire TCP/IP stack. The DVIPA address is activated at TCP/IP stack initialization and remains associated with the stack for as long as the stack is functioning correctly. The idea is communicated best graphically. In Figure 6-3 on page 111, LPAR 1 and LPAR 2 have been altered to support multiple application-instance DVIPA.

In this figure, LPAR 1 is configured to be the stack where the DVIPA address is expected to be normally active. In other words, if both LPARs are active in the

sysplex, LPAR 1 owns the IP address 10.134.61.190. The address is always active and is unaffected by the presence, or lack of presence, of any particular applications.

In the event of a loss of the TCP/IP stack (for example, an abnormal termination or a manual shutdown of the task), the TCP/IP stack configured in LPAR 2 automatically activates the 10.134.61.190 DVIPA.

Taking a closer look at the VIPABACKUP statements, there are the numbers 50 and 25 as the second parameter on LPARs 2 and 3. This is the backup host's rank. There can be more than one backup host for multiple application instance DVIPA. This rank can be used to control which backup TCP/IP stack should be activated first after the primary stack loses the address. The higher this number, the greater its priority for being used as a backup.

If a backup stack should fail, then the backup stack with the highest rank number would activate the DVIPA. In Figure 6-3 on page 111, LPAR 2 is the first host to take over LPAR 1's DVIPA. If both LPAR 1 and LPAR 2 should fail, then LPAR 3 would take over 10.134.61.190.

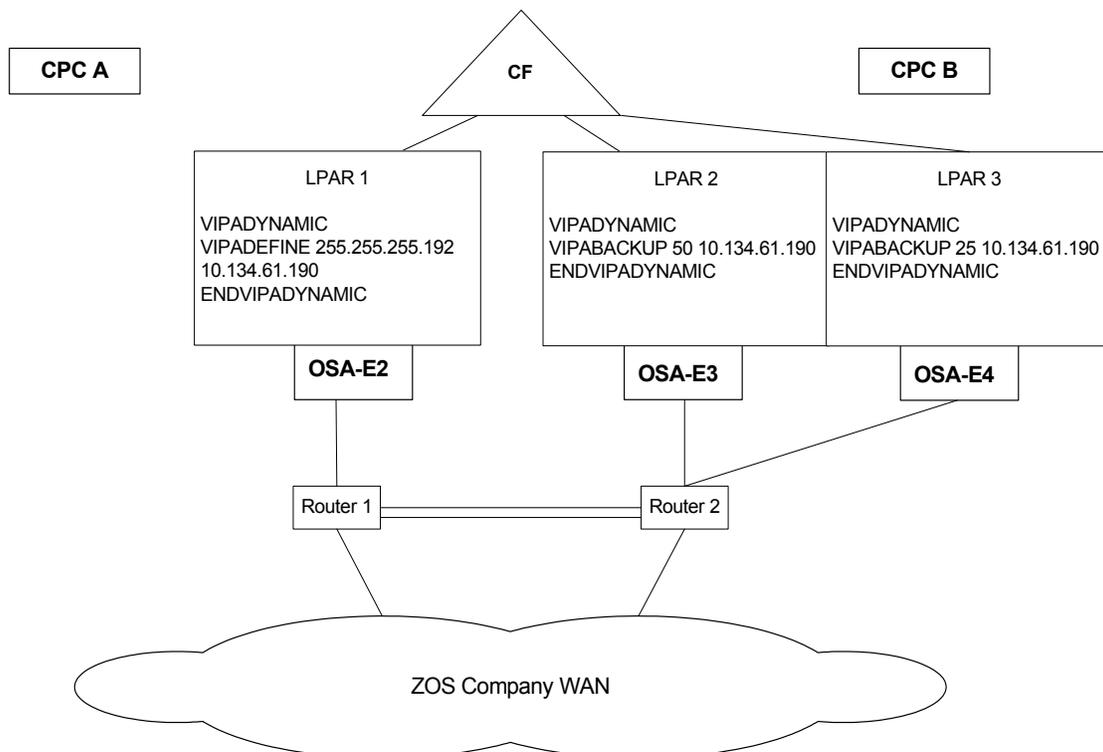


Figure 6-3 Multiple application-instance DVIPA

Implicit in unique application-instance DVIPA is the requirement that an operator (or a system automation product) be available to start the new application on the alternate LPAR when the original LPAR, TCP/IP stack, or application fails. This is not the case with multiple application-instance DVIPA.

With multiple application-instance DVIPA, if the LPAR or TCP/IP stack fails, the IP address is automatically started on one of the backup LPARs. The application could already have been running on both the primary and backup systems. Presumably the application has not used a bind() to a specific address. When TCP/IP automatically activates the multiple application-instance DVIPA on the backup host, the application is automatically available at this newly activated address.

### 6.3.3 Why DVIPA?

Having explained the details of DVIPA, let's take a moment to explain why it might be beneficial, from a network perspective.

**Availability** DVIPAs allow servers to be made available independently of hardware or software failures. This can be done dynamically by TCP/IP or even by a system automation product.

**Single image** DVIPA allows multiple LPARs to appear to be a single, highly available network host. Because DVIPA movement is automatic, end users and clients might never know a DVIPA address movement has occurred.

**Application movement** With DVIPA, applications can be seamlessly moved from one LPAR to another, allowing a virtualization of the application itself.

## 6.4 Dynamic cross-system coupling

In 6.2.1, "Cross-system Coupling Facility (XCF)" on page 105, it was noted that the XCF links can carry IP traffic. With DVIPA, XCF was used to communicate signalling information such as when a new instance of an IP address has been dynamically activated on a TCP/IP stack within the sysplex. The next step in the progression of utilizing a sysplex's capabilities is to use XCF to carry IP traffic.

### 6.4.1 VTAM and XCF

In order for TCP/IP to take advantage of inter-LPAR communication, VTAM must already have cross-system Coupling Facility (XCF) communications active.

VTAM uses XCF to establish a common membership group, ISTXCF, that TCP/IP uses when communicating over dynamic XCF connections.

## 6.4.2 Establishing the links

When dynamic XCF is functional within a sysplex, a point-to-multipoint network is established among all participating LPARs. Each host in the sysplex has a direct connection to any other host in the same sysplex.

Within the TCP/IP profile data set, there is only one configuration option required, DYNAMICXCF. This option falls within the IPCONFIG statement group, as shown in Example 6-2.

*Example 6-2 DYNAMICXCF option of IPCONFIG statement*

---

```
IPCONFIG
  DYNAMICXCF 192.168.80.1 255.255.255.0 2
```

---

The definition in Example 6-2 would cause the TCP/IP stack to create a link using the IP address of 192.168.80.1 within the sysplex subnetwork. This IP address would be directly reachable on the sysplex point-to-multipoint network by any other TCP/IP stack in the sysplex that also has DYNAMICXCF coded. Each TCP/IP stack would code a unique IP address within the same subnetwork.

When a TCP/IP stack becomes active in the sysplex and this stack has DYNAMICXCF coded, the following sequence of events occurs internally within the TCP/IP stack:

1. A DEVICE statement for this stack's XCF device is automatically generated.
2. A corresponding LINK statement is automatically generated.
3. A HOME statement entry using the DYNAMICXCF IP address is added to the active HOME list for the stack.
4. The device is started.

If a TCP/IP stack does not have DYNAMICXCF coded, it does not participate in the dynamic XCF communications. In other words, both end points must code DYNAMICXCF in order for a link to be established.

**Note:** All dynamic XCF statements must use a common network ID. This is because they form a point-to-multipoint network. Refer to 2.4.2, "Network types" on page 29 for more information about this topic.

### 6.4.3 More than just XCF

There are two special instances of dynamic XCF links that deserve further discussion.

If available, TCP/IP uses a HiperSockets link in preference to a Coupling Facility link. The reason is speed—HiperSockets is faster. For more information about HiperSockets, see 3.8, “HiperSockets” on page 63.

Dynamic XCF also functions within an LPAR when more than one TCP/IP stack is active in the same LPAR. The link generated is referred to as a *samehost link*, which corresponds to a device type of IUTSAMEH.

Consequently, the DEVICE and LINK statements generated by the TCP/IP stack vary depending upon whether a HiperSockets link is available and whether the link is within or between LPARs.

Figure 6-4 on page 115 illustrates a simplified network layout showing the dynamic XCF definitions for a sysplex. Each TCP/IP stack has a unique IP address to represent itself in the dynamic XCF subnetwork. The dynamic XCF IP addresses of each participating TCP/IP stack are all in the same subnetwork.

When a new LPAR is added to the sysplex, assuming it is configured appropriately, it automatically joins the sysplex and activate its IP address within the dynamic XCF network.

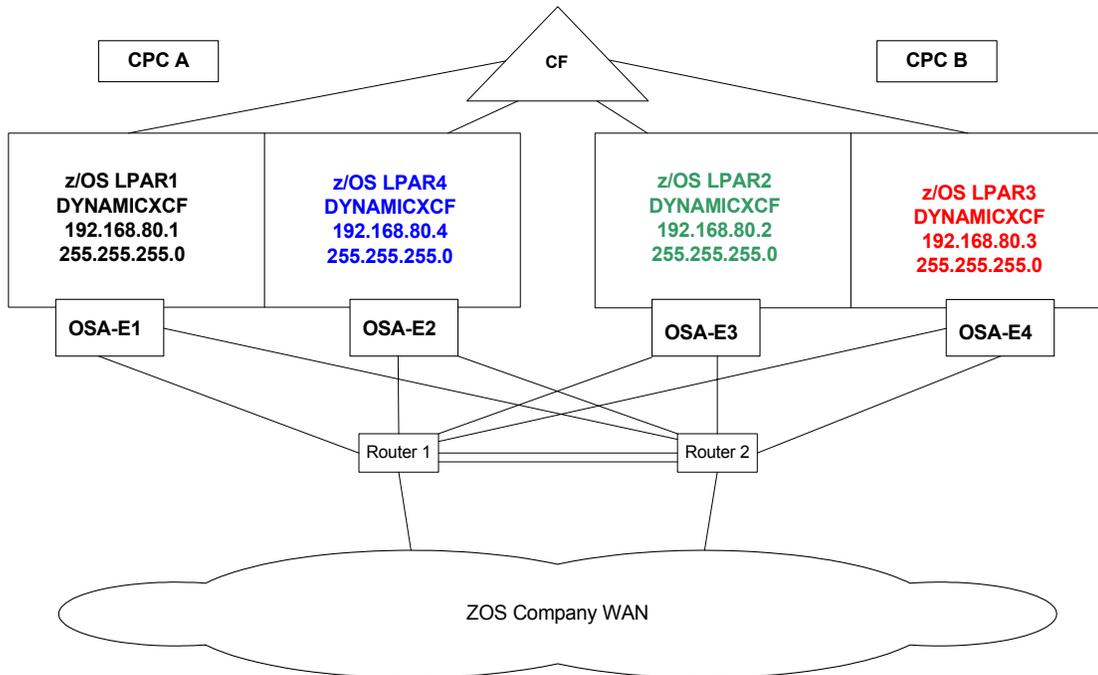


Figure 6-4 Dynamic XCF network sample

## 6.5 Sysplex distributor

IP address availability in z/OS can be viewed as a progressive evolution. Static VIPA eliminates the problem of an IP address being associated with a single networking hardware point of failure. Dynamic VIPA allows us to move such IP addresses in event of an application, TCP/IP stack, or LPAR failure. Next, dynamic XCF provides automatic generation of links to provide an automated IP layer of connectivity within the sysplex. But, what about taking advantage of the workload balancing mentioned earlier?

Sysplex distributor can be viewed as a continued evolution of connectivity improvements. It is a combination of the high availability features of DVIPA and the workload optimization capabilities of WLM.

The implementation has one significant difference: instead of all participating hosts being effectively equal, LPARs can be given specific roles to play. When combined with WLM, the overall effect on availability is exceptional.

Another change with sysplex distributor is that the distribution is possible only with TCP connections. Other layer 4 protocols are not supported.

### 6.5.1 What is a tightly-coupled host to do?

With sysplex distributor, the LPARs in the sysplex are assigned roles. A TCP/IP stack in one LPAR is given the role of being a front-end host. It receives inbound connection requests and redirects them to a specific target back-end TCP/IP stack. Other LPARs run target TCP/IP stacks that ultimately function as the real endpoint of communication.

**Note:** In the context of sysplex distributor, multiple TCP/IP stacks in a single LPAR can be functionally ignored. Consequently, to simplify our discussion, the term “host” is used to denote a LPAR containing a single TCP/IP stack.

Sysplex distributor requires that a choice be made as to whether a host is to function as a distributor, a target, a backup distributor, or some combination.

Things are about to get a little bit more complicated. So let’s define some simple terminology.

- |                          |   |
|--------------------------|---|
| <b>Distributing host</b> | The designated contact (point of entry) for the sysplex. It is the normal owner of the IP address that clients out in the network use to connect to the sysplex.  |
| <b>Target host</b>       | A host within the sysplex to which a distributing host can redirect a connection request. The target host must be running an instance of the target application. For example, if a client wants to connect to FTP, then there must be an FTP server running on the target host to which the session is distributed. |
| <b>Backup host</b>       | A host that is designated as a backup in the event that the distributing host should malfunction. The backup host takes over the IP address of the distributing host when required. There can be more than one backup host.   |
| <b>Combinations</b>      | A distributing host can also be a target host; some sessions are distributed to itself. A target host can also be a backup distributing host. The idea here is to not waste an LPAR just because it is assigned a certain role.   |

## 6.5.2 Sysplex distributor in action

With these options and flexibility, there are a considerable number of configurations possible. The layout in Figure 6-5 on page 117 is just one simple possible scenario.

For example, the only application shown as being available on all target hosts is FTP. Such a sysplex might be functioning as a very busy FTP server farm, but it presents a single IP address to all incoming FTP clients. Many other applications could be added to these hosts as target applications.

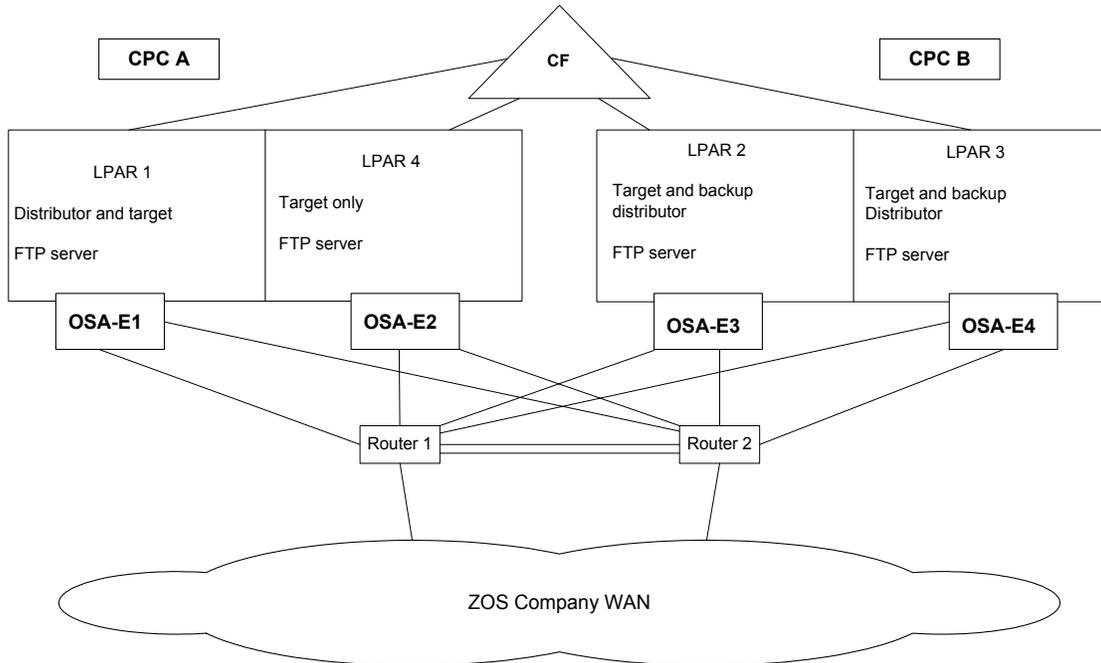


Figure 6-5 Distributed DVIPA, sysplex distributor

In Figure 6-5, there are four LPARs being used for sysplex distributor. LPAR 1 is a distributing host, and LPAR 2 is its backup. All LPARs are configured as targets for connections. The target application, of course, is the z/OS FTP server.

### **TCP/IP definitions**

Since sysplex distributor requires dynamic XCF, the first configuration step is to enable dynamic XCF on each host. All these configuration statements belong in the TCP/IP profile data set. The same definitions used in Figure 6-4 on page 115 are used here. The dynamic XCF IP addresses are used by the distributing TCP/IP host when it redirects packets to a target host.

In order to balance the connections to the appropriate targets, TCP/IP is going to need to be talking to the workload manager. By coding SYSPLEXROUTING in the IPCONFIG section of the TCP/IP profile, we can ensure that WLM is consulted for connection distribution advice.

If LPAR 1 is to function as a distributor, the next thing to be defined is the IP address it uses. Again, building on our earlier discussion, we'll use the same definitions as we used in Figure 6-3 on page 111. This means that 10.134.61.190 is the IP address a client on the network would use to connect to the sysplex.

At this stage, you might notice, the only new statement discussed is the SYSPLEXROUTING statement in order to involve WLM. But how do we go about getting the TCP/IP stack in LPAR 1 to function as a distributing host? For this, we need to add a VIPADISTRIBUTE option within the VIPADYNAMIC block of statements; see Example 6-3.

*Example 6-3 LPAR 1 sample configuration statements*

---

```
IPCONFIG SYSPLEXROUTING DYNAMICXCF 192.168.80.1 255.255.255.0
VIPADYNAMIC
VIPADDEFINE 255.255.255.192 10.134.61.190
VIPADISTRIBUTE DEFINE SYSPLEXPORTS 10.134.61.190 PORT 21 DESTIP ALL
ENDVIPADYNAMIC
```

---

The VIPADISTRIBUTE statement identifies DVIPA address 10.134.61.190 as an IP address to be distributed. Only connections for port 21 is distributed. And, those connections are only distributed to target hosts that have an active listen on port 21 (allowing for application failure or seamless application take down). The destination IP addresses are all active dynamic XCF IP addresses within the sysplex.

The SYSPLEXPORTS parameter on the VIPADISTRIBUTE statement is used to manage the assignment of ephemeral ports among the LPARs.

What about the other hosts in this sample? Do they need to have any definitions changed? For our purposes, the answer is no. Returning to Example 6-3, LPARs 2 and 3 are both backup LPARs for 10.134.61.190.

In event of a failure in LPAR 1, LPAR 2 would take over the DVIPA. In addition, it automatically inherits the distribution capabilities and characteristics that are defined for LPAR 1. If both LPAR 1 and LPAR 2 fail (either at the same time or in sequence), LPAR 3 would become the distributor of 10.134.61.190.

When the problem is recovered and LPAR 1 is brought back into the sysplex, it would non-disruptively take back the role as the distributor.

## Walking through distributed connections

Figure 6-6 on page 119 illustrates two users connecting into a sysplex with distributed connections. To simplify the illustration, only three LPARs are displayed.

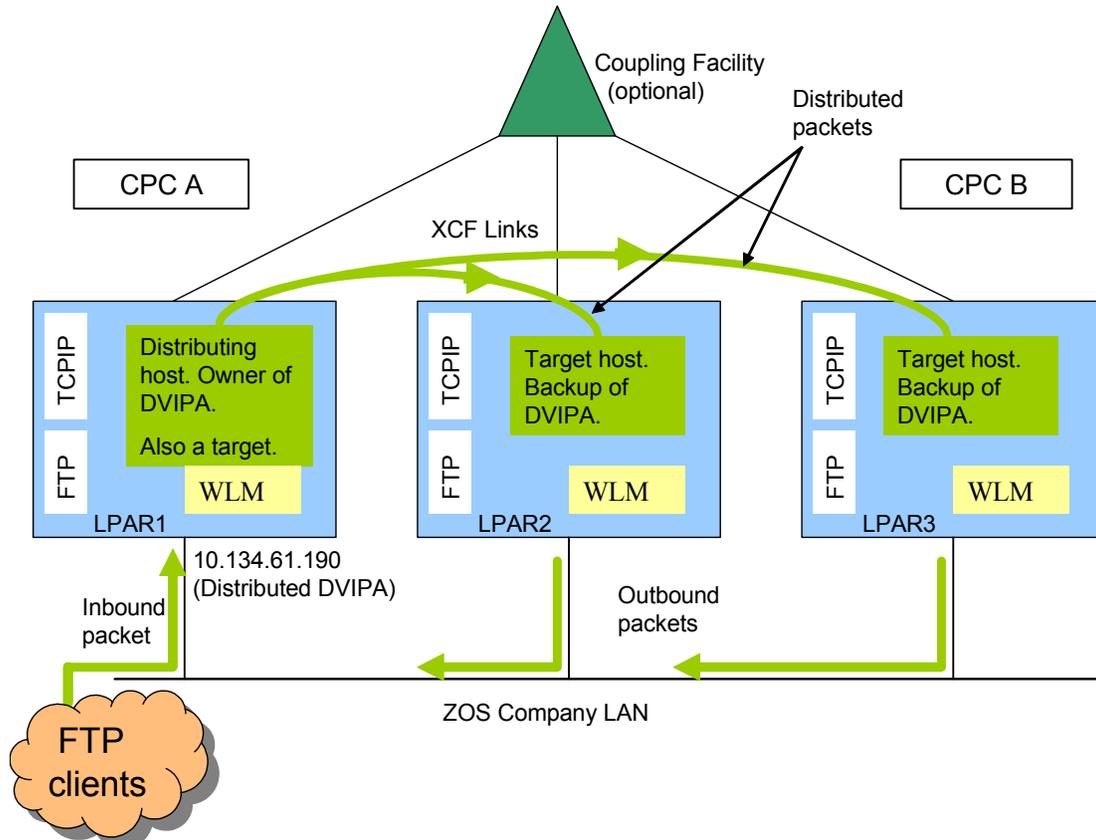


Figure 6-6 Distributed connections

Both users are using FTP clients. Here is the sequence of events as they transpire for the desktop workstation:

1. The FTP client issues a connection request to 10.134.61.190.
2. The request is received at LPAR 1.
3. WLM indicates that LPAR 2 should handle this connection.
4. The connection request is redirected (forwarded) to LPAR 2 over a dynamic XCF link.

5. LPAR 2's FTP server responds to and accepts the connection request. The source IP address used is 10.134.61.190 (otherwise the connection would fail at the FTP client end).
6. The FTP session continues as usual.

Next, a notebook user starts an FTP client and attempts to connect to 10.134.61.190. This time, in step 3, WLM indicates that the next connection should not be redirected at all. Instead, this connection is completed without any distribution performed.

Some questions might pop into your head at this time. You may be thinking, "Shouldn't the FTP environments have to be the same on all LPARs to ensure that users always appear to be connecting to the same server?" Yes, you're correct.

And this can be accomplished in a couple of ways:

- ▶ z/OS can share DASD (disks) within the sysplex so that each FTP server accesses the same environment.
- ▶ If it is a retrieve-only FTP environment (no PUT of files allowed), then the file systems could readily be duplicated within each LPAR's environment.

Another question you might have depends upon your knowledge of how FTP works. Each time a data transfer request (or even a `dir` or `ls` command) is made in FTP, a separate and new connection must be established. FTP has two ways it can establish the data connection: by using a `PORT` command or a `PASV` command (for details, see RFC 959).

### ***PORT command***

The `PORT` command method of establishing the data connection requires the FTP server to send a connection request back to the client. Such a scenario is not complicated by its occurring within a distributed sysplex because an outbound connection does not need to use the distributor. The connection request heads outbound over the appropriate link directly to the host where the FTP client is running.

### ***FTP PASV command***

In contrast, an FTP `PASV` command requires that the data connection is established by the client sending a connection request to the server. When this data connection arrives at the distributing host, how does it know which FTP server owns the existing control connection? The answer is that the distributing host is state-aware, in much the same fashion as a stateful firewall. The distributing host keeps track of any inbound TCP connections. In addition, the distributing host is also made aware of target hosts establishing outbound

connections, though it needs to keep track only of outbound connections that use the distributed address as a source address.

One final thing to note in Figure 6-6 on page 119 is that responses from target servers do not need to return through the distributing host. Instead, they can follow a direct return path.

## Controlling distribution

z/OS can function as a server for thousands of different large scale applications, like CICS, IMS, and all of the well-known IP servers. These applications are often mission-critical and experience workloads in the range of tens of thousands concurrent users. An organization has guidelines for the service level that is appropriate for a given service. (Sometimes, these guidelines are formalized in a service level agreement.)

In such a context, some hosts in the sysplex would best be left with more or less overall workload than others. To this end, the distribution numbers provided by WLM can be modified by the z/OS policy agent. By modifying WLM's numbers with the policy agent, workload can be shifted according to the overall requirements of the organization.

**Note:** A full description of the policy agent is beyond the scope of this text, but you should know that the policy agent is a z/OS application that influences such networking functions as:

- ▶ TCP/IP quality of service (TCP session control)
- ▶ Intrusion detection services (recognition of dangerous IP traffic as it arrives at the z/OS host). See Chapter 12, “Network security” on page 283 for more information.
- ▶ Sysplex distributor.
- ▶ Packet filtering, Application Transparent - TLS, and Virtual Private Networks, all of which are explained in Chapter 12, “Network security” on page 283.

The policy agent influences these environments based upon explicitly coded policies. These policies can be retrieved from LDAP servers, which can be accessed from a host on the network (allowing a centralized repository of policies shared among all platforms, not just z/OS).

There are other, less complex, methods of controlling the distribution of connections with sysplex distributor. The `VIPADISTRIBUTE` statement in Example 6-3 on page 118 did not code the distribution method, so the system

used the default (DISTMETHOD BASEWLM). That is, the same effect could have resulted from coding:

```
VIPADISTRIBUTE DEFINE SYSPLEXPORTS DISTMETHOD BASEWLM 10.134.61.190 PORT 21
DESTIP ALL
```

The DISTMETHOD BASEWLM option means the distribution method uses WLM. If we change this option to DISTMETHOD ROUNDROBIN, sysplex distributor does not consult the Workload Manager at all. Instead, it does simple round robin distribution: connections are distributed continually to consecutive hosts.

Beginning with z/OS V1R7, sysplex distributor can control distribution based upon finer-grained data from WLM. In particular, sysplex distributor can take into account whether a specific application is meeting its performance goals as defined to WLM and can keep track of the target server's response time. Faster responding servers receive more connections. Instead of coding DISTMETHOD BASEWLM in the sample above, DISTMETHOD SERVERWLM would be used for this.

### 6.5.3 Moving distribution to the network

Cisco Systems has a load balancing and connection distribution solution called Content Switching Module (CSM). CSM provides layer 4 to layer 7 routing and failover control based upon message (packet) content and the health of target servers. On z/OS, the health of a sysplex is communicated using the Load Balance Advisor, LBA. The LBA communicates health data to CSM. The CSM then makes decisions on which server to forward workload to. The end result is very similar to the way sysplex distributor functions, but the implementation is significantly different. In addition, the CSM can communicate with non-z/OS platforms.

Instead of the sysplex distributor node being the target for inbound connections, client connection pass through the CSM. The CSM handles each incoming session and distributes it to the appropriate target host.

### 6.5.4 Problem detection and recovery in the cluster

Continuing on our progression of continual improvement, sysplex distributor provides a tightly-coupled cluster of independent hosts capable of dynamically managing availability of a single IP address. When everything is working as designed, the system is as effective as can be imagined.

However, everything does not always work as designed. What happens if TCP/IP, or an entire LPAR, is running, but not exactly running effectively? For example, what happens if a critical component is unresponsive or unavailable? The handling of such a situation is often referred to as *system autonomies* or *autonomic computing*.

Each TCP/IP stack in the sysplex automatically monitors its own health (and the health of its own LPAR). Here is a high-level overview of what is monitored:

- ▶ VTAM address space availability.
- ▶ Communications Storage Manager (CSM) storage shortage. CSM is a component of VTAM that handles real storage memory usage. For more information on CSM, see 1.7.1, “z/OS Communications Server” on page 15.
- ▶ XCF connectivity.
- ▶ OMPROUTE availability (see 6.6, “Routing in a sysplex” on page 123).
- ▶ TCP/IP stack responsiveness (as checked by XCF once TCP/IP has registered with XCF).
- ▶ Connection success rate (are new connections successfully established?). This includes monitoring of whether a backlog exists and the connectivity between the distributor and a target, as well as between a target and its client.

The statement to activate this capability is shown in Example 6-4.

*Example 6-4 Sysplex recovery*

---

```
GLOBALCONFIG  
SYSPLEXMONITOR RECOVERY
```

---

These resources are by default monitored on 60-second intervals. If a resource is deemed to be unavailable, the RECOVERY option causes the TCP/IP stack to remove itself from the sysplex. After the problem has been recovered, the TCP/IP stack can automatically rejoin the sysplex (facilitated by having coded GLOBALCONFIG AUTOREJOIN in the TCP/IP profile data set).

## 6.6 Routing in a sysplex

You might wonder how to keep on top of the routing requirements of these shifting IP addresses and shifting TCP/IP stacks.

What sort of routing solution can meet the changing needs of a DVIPA? We can begin by eliminating static routing (which is the type of routing illustrated in Chapter 5, “TCP/IP on z/OS” on page 81). Static routing, under many circumstances, requires manual intervention if a DVIPA moves. Manual intervention would defeat the value of the sysplex’s dynamic operations.

Looking at dynamic routing protocols, RIPv1 and RIPv2 should also be eliminated. The problem is that RIP protocols have a slow rate of convergence (up to 3 minutes). *Convergence* is the rate at which a network routing change is recognized and adapted to within a network. The value of a DVIPA is greatly

diminished if it takes the routing layer several minutes to adapt to an IP address being moved.

The next choice of dynamic routing protocol is Open Shortest Path First, or OSPF, and it is the protocol recommended to support DVIPAs in a sysplex environment. With OSPF, when a DVIPA is moved to another host, convergence is both fast and automatic. Other hosts in the network can rapidly become aware of the new location of the DVIPA.

For more information on routing protocols, see Chapter 2, “Network layers and protocols review” on page 21.

### **6.6.1 Open Shortest Path First (OSPF)**

Open Shortest Path First (OSPF) has become an industry standard for dynamic routing within a given network. It is standardized most recently within RFC 2328, but OSPF standards seem to dapple the RFC landscape. Usually, the term OSPF is used to describe OSPF Version 2.

On z/OS, the OSPF protocol is implemented by the OMPROUTE server. The OMPROUTE server is actually a z/OS UNIX daemon. At the time of writing, OMPROUTE implements the OSPF Version 2 protocol as outlined in RFC 1583, as well as almost all of the features from RFC 2328. The important part is that OMPROUTE is completely compatible with adjacent routers that are capable of handling OSPF communications.

#### **OSPF requirements in a cluster**

The actual configuration of OSPF on z/OS, or any other host, is relatively straightforward. The difficult part is the network topology planning. From a network design viewpoint, OSPF is incredibly flexible, which translates to incredibly complex. Designing a well-planned network is a challenge.

The good news is that once an OSPF network has been designed, the actual configuration and execution of OMPROUTE is a straightforward task. However, there is one specific recommendation that we should discuss.

Even though OMPROUTE is a fully functional OSPF router, the z/OS operating system is not designed to be a backbone router operating system. So, a z/OS host (and in our case, a sysplex) should be implemented with the intention of having its dynamic routing requirements minimized. In the case of a sysplex, the ideal is to define the area encompassing the sysplex as a stub area. This is depicted in Figure 6-7 on page 125.

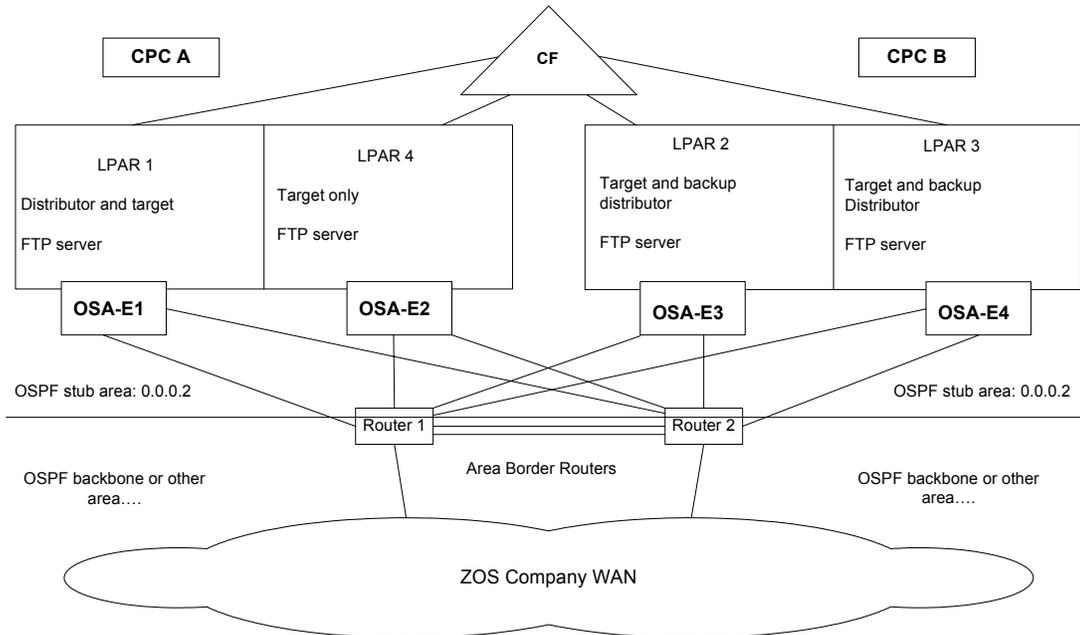


Figure 6-7 OSPF stub area for sysplex

The area above the dotted line is area 0.0.0.2 (OSPF requires all router interfaces to be identified as being within an area; note that the number 0.0.0.2 is not significant—the area only needs to be unique).

What exactly is a stub area? The OSPF protocol defines a given routing area (as identified by the area number) as either being a transit area or a stub area. A *transit area*, as its name suggests, is an area that can have traffic passing through it. A *stub area* is nothing more than a network dead-end. Packets can flow into and out of a stub area, but traffic does not travel through it.

Why is this recommended for the sysplex? Here are some characteristics of a stub area:

- ▶ A stub area router does not receive updates about the network beyond the stub area.
- ▶ A stub area does not have any inter-area traffic (that is, you cannot reach any other OSPF network areas by travelling through the stub area).
- ▶ Each stub area router communicates only default route and intra-area information.

The two routers shown in Figure 6-7, Router 1 and Router 2, are the routers that straddle two areas. These routers are referred to as *area border routers*, or ABRs.

When updates from other areas arrive at an ABR, the information is not forwarded to the stub area.

Ultimately, an OSPF stub area allows the sysplex to be connected dynamically to the outside network and to dynamically reconfigure as appropriate within the sysplex. At the same time, the stub area minimizes the routing overhead required within the sysplex itself.

### **Rejoining an active sysplex**

It was said before that communication within the sysplex is, as far as TCP/IP is concerned, effectively instantaneous. This means that when a TCP/IP stack joins the sysplex, its status is reflected immediately to all TCP/IPs participating in the sysplex. Yet, there is no guarantee that OMPROUTE is started immediately.

In fact, the TCP/IP started task can autostart a very large number of TCP/IP applications, of which OMPROUTE is only one. Obviously, OMPROUTE needs to build a routing table for TCP/IP in order for network communication can occur. TCP/IP can be configured to ensure that it does not join (or rejoin) the sysplex until OMPROUTE itself has been started. This reduces the possibility of a connection being routed to a host that doesn't yet have connectivity to the network.

## **6.6.2 Network interface card**

The network interface cards used in all the diagrams in this chapter have been OSA-Express cards. There is so much capability built into an OSA-Express card that it warrants a significant portion of Chapter 3, "Hardware connectivity on the mainframe" on page 45. It is appropriate though, to mention here some of the cooperation that occurs between TCP/IP and OSA-Express, and how this can effect availability within a sysplex environment.

The first point to note is that an OSA-Express card (when configured appropriately) automatically receives a copy of all IP addresses that are active within a TCP/IP stack. When a DVIPA is added to or removed from a TCP/IP stack, the OSA-Express card is immediately informed of the change.

The second point of note is that for simplicity's sake, the figures in this chapter have all shown a single OSA-Express card attached to each individual LPAR. Each OSA-Express card, however, has two physical connectors (adapters) available. And, an OSA-Express card can be shared with more than one LPAR (within a CPC, of course). As a result, most implementations of an OSA-Express card involve sharing the OSA-Express card between two LPAR's.

In Figure 6-8 on page 128, the OSA adapters have been pulled away slightly from the LPARs in order to show the cross-adapter redundancy of sharing OSA

adapters. The presumption here is that all adapters are connected to the same LAN segment and the same subnetwork.

If we imagine that the OSA-E2 card were to fail, we can readily see that both LPARs would maintain complete connectivity to the LAN using OSA-E3. What is not so apparent is the gratuitous ARP processing. Gratuitous ARP processing is part of the availability features of the OSA-Express card. For more information, see the discussion of ARP takeover in 3.7.1, “Queued Direct I/O (QDIO)” on page 58.

The end result is that even though IP addresses 201.2.11.1 (LPAR 1) and 201.2.11.4 (LPAR 2) appear to be lost when OSA-E2 is lost, they are immediately revived in the OSA-E3 card by a gratuitous ARP updating the MAC addresses of the lost IP addresses.

In other words, network connectivity is re-established by making changes at layer 2 (link layer) instead of layer 3 (IP layer). The routers in the network would effectively not be involved and routing tables remain unchanged.

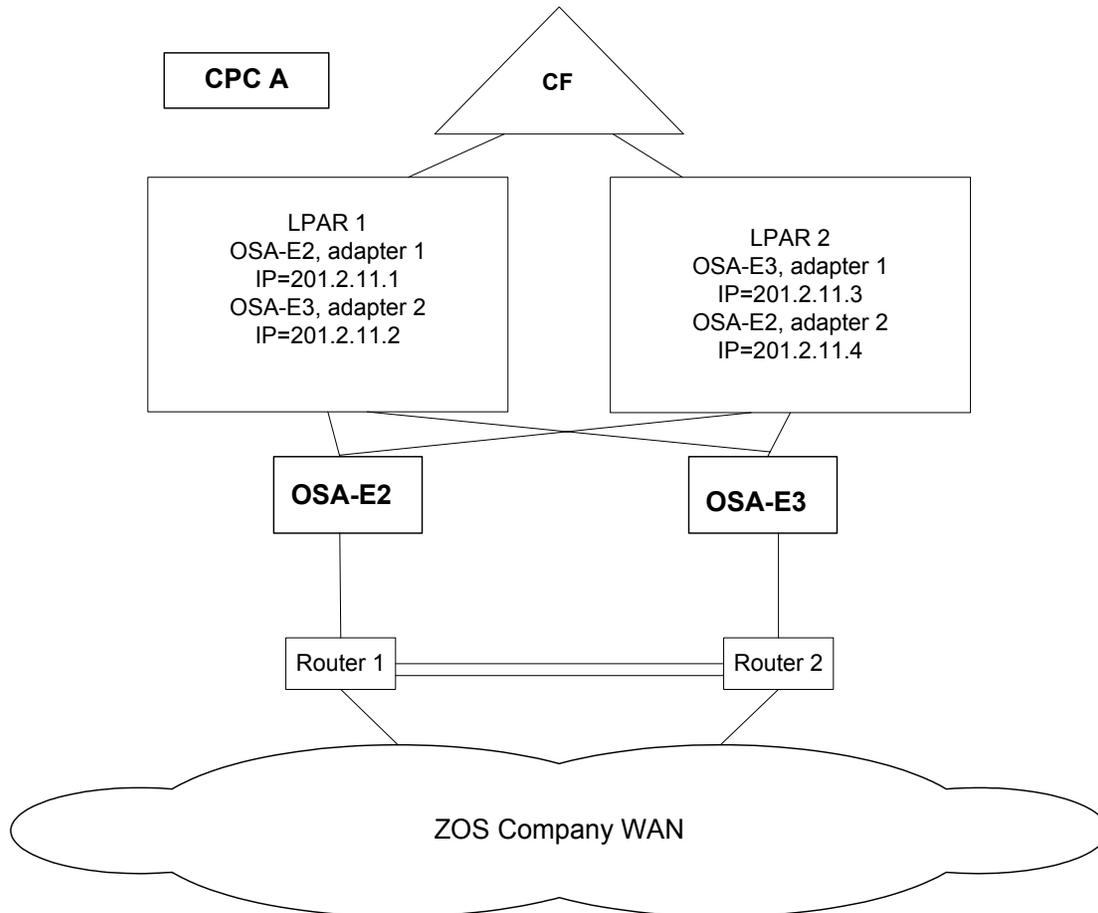


Figure 6-8 Shared OSA adapters in a sysplex

## 6.7 Summary

A sysplex is a cluster of tightly-coupled independent instances of the z/OS operating system. The internal communications within a sysplex are facilitated by the Coupling Facility. TCP/IP takes advantage of the Coupling Facility and Workload Manager to optimize availability and load balancing in a sysplex.

Availability is enhanced by the ability to dynamically move IP addresses using dynamic VIPA. Sysplex distributor combines dynamic VIPA, Workload Manager, and autonomic computing to create the highest possible availability of an IP host.

Within a sysplex environment, OSPF is used to handle routing changes dynamically.

<b>Key terms in this chapter</b>		
cross-system Coupling Facility (XCF)	dynamic XCF	multiple application-instance DVIPA
Open Shortest Path First (OSPF)	Request for Comments (RFC)	stub area
sysplex	sysplex distributor	unique application-instance DVIPA
Workload Manager (WLM)		





## Part 3

# **SNA and SNA/IP implementation on the mainframe**





# Systems Network Architecture - basics and implementation

**Objective:** After completing this chapter, you will be able to:

- ▶ Describe the basic concepts of SNA subarea networking.
- ▶ Define the major components in an SNA network.
- ▶ Explain routing and addressing in an SNA subarea.
- ▶ Describe SNA flow control.
- ▶ Describe VTAM definitions for subarea nodes.

## 7.1 What is Systems Network Architecture (SNA)?

Systems Network Architecture (SNA) is a data communication architecture established by IBM to specify common conventions for communication among the wide array of IBM hardware and software data communication products and other platforms. Among the platforms that implement SNA in addition to mainframes are IBM's Communications Server on Windows, AIX, and Linux, Microsoft's Host Integration Server (HIS) for Windows, and many more.

The way in which products internally implement these common conventions can differ from one product to another, but because the external interface of each implementation is compatible, different products can communicate without the need to distinguish among the many possible product implementations.

SNA products recognize and recover from loss of data during transmission, use flow control procedures to prevent data overrun and avoid network congestion, identify failures quickly, and recover from many errors with minimal involvement of network users. SNA products also increase network availability through options such as the extended recovery facility, backup host, alternative routing capability, and maintenance and recovery procedures integrated into workstations, modems, and controllers.

### 7.1.1 History

In 1974, IBM introduced its Systems Network Architecture (SNA), which is a set of protocols and services enabling communication between host computers (IBM mainframes) and peripheral nodes, such as IBM's dedicated hardware boxes, the 3174 controller for 3270 type displays and printers, controllers for the retail and finance industry, and more. The mainframe subsystem that implements SNA was named Virtual Telecommunication Access Method (VTAM).

The robustness of the SNA protocol, the IBM hardware, and the transaction management infrastructure software supplied by IBM (CICS and IMS) made SNA the dominant protocol in the Fortune 1000 companies.

In order to understand the rationale of the many functions and services in SNA, you must understand the computing environment at that time. Prior to 1974, data processing was batch-based. *Batch* means that data was recorded on paper, usually on predefined templates, and was keyed into media (like punched cards) readable by the computer system. The computer department executed various programs on the data. The final result was a printed report.

Around 1974, *transaction processing* was introduced. People used terminals to key in the data directly and receive the output for their inquiry instantaneously. To implement transaction processing, networking infrastructure was put in place.

The carriers at that time were geared to supply voice services rather than data service, so communication lines were slow and unreliable (in the range of 9600 bits per second).

The human ear can tolerate small errors in telephone lines, but computers cannot. Even a missing bit, or an extra bit, in a data communication line can be catastrophic. Try to imagine what might happen to your bank account if the ATM you use receives a garbled message.

In the early 1970s, computer memory was a scarce and expensive resource. Devices with 16 KB of memory were common in the computer industry. These devices were slow compared to the CPU speeds we see today.

IBM had to address the limitation imposed by the communication lines and networking hardware, and developed a robust protocol that would guarantee the integrity of the messages.

## 7.1.2 What you need to know about SNA today

During the 20-year period when SNA was the primary networking method, many CICS and IMS application programs were developed and put in place. The application programming interface (API) of these application programs is heavily dependent on the underlying protocol, SNA.

It is apparent that TCP/IP is the dominant networking protocol now and for the foreseeable future. Today, new applications use state-of-the-art programming techniques, like Java™ and HTTP, but it will take many years until all SNA applications disappear. Why is that so?

A networking application is dependent on the communication protocol it uses. Every protocol provides an application programming interface (API). TCP/IP's API is called *socket programming* and SNA has its own API. Migrating a networking application from one protocol to another (that is, from SNA to TCP/IP) requires replacing the calls to the API. Business managers are reluctant to invest in protocol conversion only for the sake of changing the underlying protocol without introducing new functions and improvements.

More importantly, in the past 30 years businesses have invested a tremendous amount of labor and money in developing SNA applications. It is estimated that the investment made in CICS and IMS applications is in the range of 20 trillion US dollars. Considering the investments in SNA applications, these programs will be used for many years. To recode these applications as TCP socket applications is often impractical and cost-prohibitive. Besides, alternatives exist.

IBM introduced new technologies to help businesses preserve the investment in SNA and use IP as the protocol for connecting SNA computers. The technology

is known as SNA/IP (“SNA over IP”) integration and it is described in detail in Chapter 9, “SNA/IP implementation” on page 211. The two endpoints, the SNA application in the mainframe and the SNA application in the remote location (branch, store), remain unchanged, thereby preserving the investment in SNA.

Because SNA applications will exist for years to come, someone has to care for SNA definitions, problem determination, recovery, business continuity procedures, and many other tasks. These tasks are the responsibility of the mainframe networking systems programmer who needs to know in depth the architecture and how to implement SNA on various hardware and software platforms.

## 7.2 The evolution of SNA

Over the years, SNA has evolved to accommodate new technologies and adapted to the changes in data communication.

Today, there are two implementations of SNA: subarea networking and Advanced Peer-to-Peer Networking (APPN):

- ▶ Subarea networking

Subarea networking was the initial implementation of SNA that defined mainframe-based hierarchical networks in which every resource and SNA route had to be predefined. In the initial implementation of SNA, adding resources or changing SNA routes necessitated the shutdown of parts of the network.

- ▶ Advanced Peer-to-Peer Networking (APPN)

To address the deficiency of the static nature of subarea SNA, IBM introduced an SNA-based peer network, with no hierarchical relations, and with dynamic definition of resources.

At a later stage, APPN was enhanced with the introduction of High Performance Routing (HPR) and SNA/IP, which, as its name implies, is a high performance routing protocol that can be optionally exploited by APPN.

Neither subarea networking nor APPN resolved a weakness related to the loss of an SNA session when a resource along the session route fails. Besides improving routing performance, HPR provides non-disruptive re-routing of the SNA session to an available alternate route. HPR also enables the integration of SNA into IP-based backbones.

Hierarchical systems are organized in the shape of pyramid, with each row of objects linked directly to objects beneath it. SNA subarea, besides implementing

the model of a hierarchical system, is centrally managed from the top of the pyramid.

Network resources in SNA are managed (that is, known and operated) from a central point of control that is aware of all the activity in the network, whether a resource is operational, and the connectivity status of the resource. The resources can send reports on their status to the control point. Based on networking and organizational requirements, a hierarchical network can be divided into sub-networks, where every sub-network has a control point with its controlled resources.

We can use an airport control tower as an example to explain the centrally-managed approach. All airplanes in the control tower sphere of control (a sub-network) are controlled and report to the control tower. The control tower also “operates” the resources (airplanes and runways) by granting landing and takeoff authorization.

In a peer network, every resource is self-contained and controls its own resources. Most of the time a networking resource in a peer network is not aware of its network peers, and learns about their existence when it starts to communicate with the peer resources.

We can use a Windows workstation as an example. We define only the local network of the workstation. The workstation can connect and exchange data with every resource it is authorized to access, as long as the physical path is available.

A national real estate franchise is good illustration of a peer network. Every local real estate office maintains the listing in its area and is not aware of the information stored in other offices. If a customer who plans to relocate asks for service from the local office, the office will call (connect to) the office in the city his customer plans to move to and get the listing from the remote location. If the customer had not made this request, the local office would not be aware of the remote office, and would learn about the remote office only when there was a need to access data that was stored remotely.

By now, you are probably asking yourself why SNA initially followed the hierarchical path and TCP/IP, which was developed at the same time, is a peer protocol. Well, the answer is that the goals of the protocols were different. TCP/IP was developed to provide collaboration between computers and data sharing. SNA was developed for central control.

In the 1980s, TCP/IP was used extensively by scientists who wanted to share research papers and ideas stored on their campus computers with academic staff around the world. IBM designed SNA for business data processing applications. The hierarchical topology of SNA matches the organizational

structure of businesses and enterprises. The most common example is a bank where the tellers in the branch require access to the bank's central database. The same paradigm is also true for the insurance and retail industry. Also, businesses that have regional offices connected to a corporate site can implement the hierarchical network model.

## 7.3 Subarea networking

The initial implementation by IBM was the SNA subarea network. This network is a hierarchical network implemented by the Virtual Telecommunications Access Method (VTAM) in the mainframe. VTAM is the software that controls communication and data flow in an IBM mainframe SNA network. VTAM resides in the mainframe and supports a wide variety of network protocols, like SDLC and LAN.

VTAM controls data transfer between channels and OSA LAN-attached devices and performs SNA routing functions.

VTAM provides an application programming interface (API) that enables the development of application programs that communicate using SNA with remote application programs or devices.

Currently, VTAM is part of Communications Server for z/OS and is called SNA services.

### 7.3.1 SNA nodes

A data communication network can be described as a configuration of nodes and links. Nodes are the network components that send data over, and receive data from the network. Node implementations include processors, controllers, and workstations. Links are the network components that connect adjacent nodes. Nodes and links work together in transferring data through a network.

An SNA node is a set of hardware and associated software components that implement network functions. Nodes differ based on the architectural components and the set of functional capabilities they implement. Nodes with different architectural components represent different node *types*.

In SNA, four types of nodes exist:

**Type 5 (T5)** A T5 node is located only in the mainframe. The software that implements the T5 node is the SNA component of the Communications Server. The SNA component in z/OS is also referred to as VTAM (Virtual Telecommunications Access Method).

- Type 4 (T4)** A T4 node is a communication controller attached to peripheral nodes through communication lines or LAN, to another communication controller through communication lines, or to a mainframe through an ESCON or a parallel channel. IBM uses special hardware and software to implement the T4 Node. The software is IBM's network control program (NCP) and the hardware is the IBM 3745 or 3746 device. The Communication Controller of Linux (CCL) is a software package that replaces the 3745 or 3746.
- Type 2.0 (T2.0)** A T2.0 is a peripheral node that attaches to the communication controller or the mainframe. T2.0 is an alias for the IBM 3174 display controller, which attaches 3270 displays and printers and is connected through a communication line to the T4 node or through a channel to the T5 node. Additional devices that implement T2.0 node are banking branch controllers and retail store controllers.
- Type 2.1 (T2.1)** A T2.1 is a peer-oriented peripheral node that attaches to a mainframe, a communication controller, or another peripheral node. A T2.1 node is called a low entry networking (LEN) node.

The links connecting two subarea nodes, either type 5 to type 5, or type 5 to type 4, or type 4 to type 4 are called *transmission groups*. For more information about connecting subarea nodes, see 7.3.4, "Connecting subarea nodes" on page 143.

**Note:** Many of the SNA node types have been replaced by up-to-date hardware and software.

The T2 node was replaced by a workstation (Windows or Unix) that implements software called "3270 emulation" and the banking and retail controller by Windows, Unix, or Linux-based servers.

The 3745 and 3746 hardware is nearing its end of life. The migration to TCP/IP in the backbone reduces the number of lines in the 3745 and 3746. The OSA and the routers can implement most of the functions of the 3745 and 3746 at much lower cost.

One of the alternatives to the 3745/3746 hardware is IBM's Communication Controller of Linux (CCL) software package implemented on the mainframe. CCL uses OSA for NCP (OSN) and routers.

After reading the previous note, most likely, you asked yourself what is the benefit of studying network technology that is not up-to-date. One of the answers is included in 7.1, "What is Systems Network Architecture (SNA)?" on page 134.

Figure 7-1 on page 140 shows the SNA nodes in a subarea network and the connections between the various nodes.

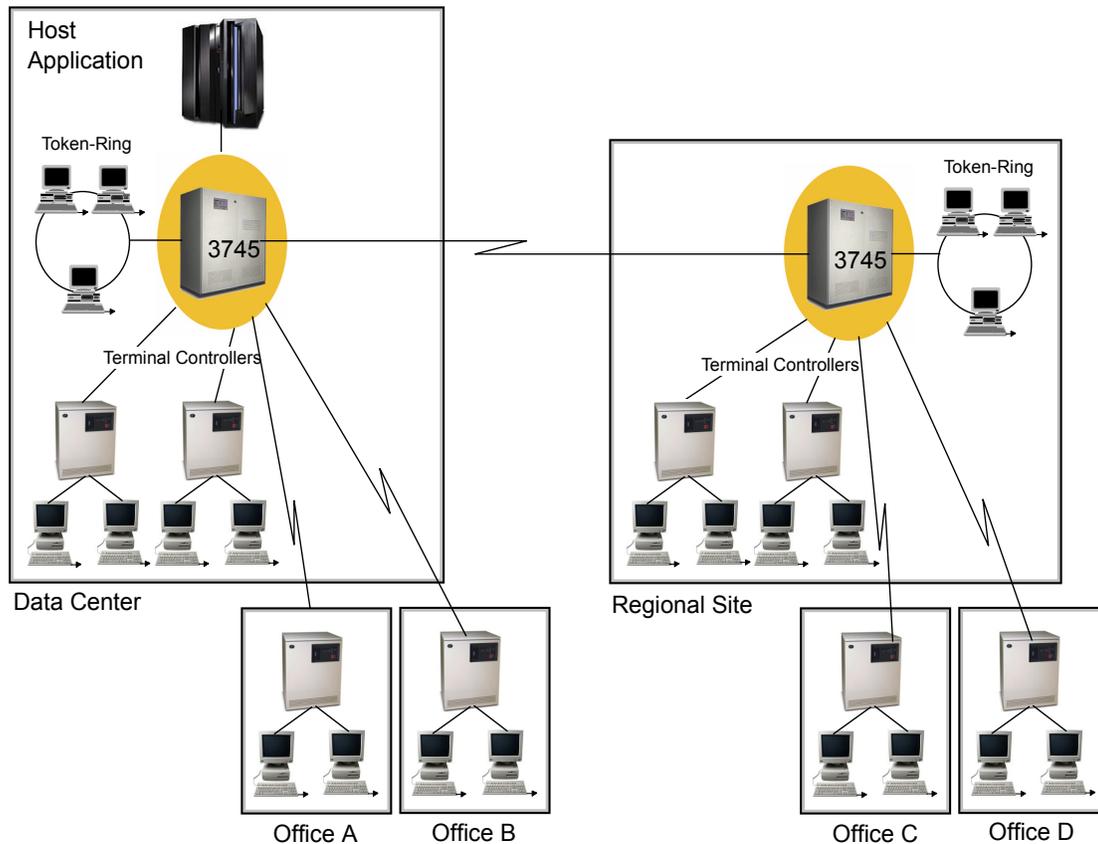


Figure 7-1 SNA nodes connectivity

Nodes that perform different network functions are said to act in different network roles. It is possible for a given node type to act in multiple network roles. A T4 node, for example, can perform an interconnection role between nodes at different levels of the subarea network hierarchy. When a T2.0 or T2.1 is connected directly to T4 node, the T4 node performs a *boundary function*. When interconnecting nodes in different subarea networks, the T4 node performs a *gateway function*.

Every T5 node in a subarea network contains a *control point*, which in general manages the network resources. Management activities include resource activation, deactivation, and status monitoring.

**Note:** APPN nodes, discussed in Chapter 8, “SNA Advanced Peer-to-Peer Networking (APPN)” on page 165, also implement a control point.

A control point's domain and the range of its capabilities depend on the type of node (APPN or subarea) in which it resides. Regardless of the node type, a control point performs the following common functions:

- ▶ Manages resources
- ▶ Monitors and reports on the status of resources

### 7.3.2 System services control point (SSCP)

A type 5 subarea node contains a *system services control point* (SSCP). An SSCP activates, controls, and deactivates network resources in a subarea network. In order to control and provide services for its subordinate nodes, an SSCP establishes *sessions* with components in the network.

For example, using a directory of network resources, an SSCP can use a session to assist an application in locating a partner and establishing a communications session. An SSCP provides the following functions:

- ▶ Manages resources on a subarea network level in accordance with the commands that network operators issue
- ▶ Coordinates the initiation and termination of sessions between applications in separate nodes within its domain, or across domains in cooperation with other SSCPs
- ▶ Coordinates the testing and status monitoring of resources within its domain

### 7.3.3 Subareas and domains

In a subarea network, every T5 and T4 node is assigned a subarea number. The subarea number has to be unique in the SNA network.

The SNA network is assigned a network identifier referred to as NETID. All the resources in the same subarea network carry the same NETID name. In the same NETID subarea network, you can have more than one z/OS system that implements the SNA protocol.

Every z/OS system with VTAM that implements SNA is referred to as a *domain*, which is an area of control. Within a subarea network, a domain is that portion of the network managed by the SSCP in a T5 subarea node. When a subarea network has only one T5 node, that node must manage all of the network resources.

A subarea network that contains only one T5 node is a *single-domain subarea network*. When there are multiple T5 nodes in the network, each T5 node may control a portion of the network resources. A subarea network that contains more than one T5 node is a *multiple domain subarea network*.

The SSCP can also set up and take down sessions with other domains through the cross-domain resource manager (CDRM). Figure 7-2 on page 143 illustrates a cross-domain network. Before applications in one domain can have cross-domain sessions with resources in another domain, a CDRM session must be established between the SSCPs of the two domains.

For a session between SSCPs to exist, VTAM must know about all cross-domain resource managers with which it can communicate. You must define to VTAM its own cross-domain resource manager and all other cross-domain resource managers in the network.

The cross-domain resource manager that represents the SSCP in your domain is called the *host cross-domain resource manager*. The cross-domain resource managers that represent the SSCPs in other domains are called *external cross-domain resource managers*.

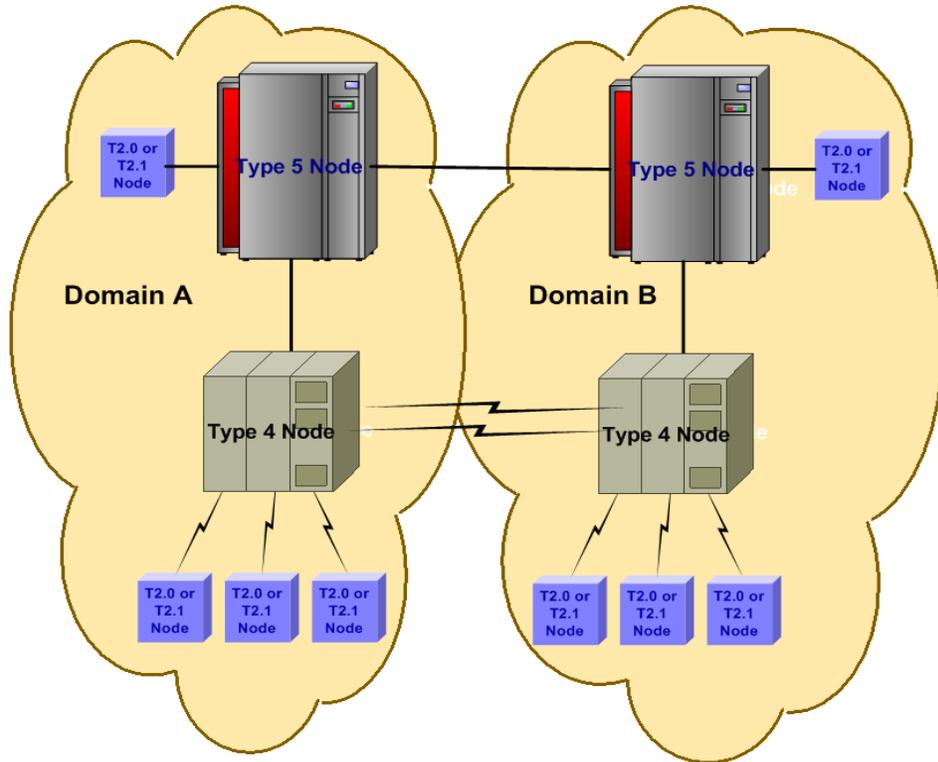


Figure 7-2 SNA subarea cross-domain network

### 7.3.4 Connecting subarea nodes

To establish a cross-domain session, that is, a session from a resource located in one domain to a resource connected in another domain, you have to have physical connections (channel, OSA, communication lines) in place to define the logical connection between the subarea nodes (T5 and T4 nodes).

#### ***Transmission groups***

A *transmission group* (TG) is a physical link or group of physical links with similar characteristics connecting adjacent nodes that is viewed as a composite unit for routing SNA messages. These multiple parallel links protect against individual link errors.

Each transmission group is identified by assigning the same number (called a *transmission group number*) to each link in the group. Links can be assigned to transmission group numbers 1 through 255. Several links between the same two

subarea nodes might have the same TG number, or the TG number might represent only one link.

### ***Explicit route***

The physical path between two adjacent subarea nodes is an explicit route. An *explicit route* (ER) is an ordered set of subarea nodes and transmission groups along a path between communicating subarea nodes, including:

- ▶ The endpoint subareas
- ▶ Any subareas between the endpoint subareas
- ▶ The transmission group used to connect each subarea pair along the route

They are explained as follows:

**Forward explicit route:** Explicit routes originating in a T5 or T4 are referred to as forward explicit routes and are numbered 0 through 15.

**Reverse explicit route:** Reverse explicit routes that terminate in a T5 or T4 must use the same set of subarea nodes and transmission groups as their corresponding forward explicit route. They are also numbered 0 through 15, but they do not have to have the same explicit route number as the corresponding forward explicit route.

### ***Logical paths***

The logical path between two subarea nodes is a virtual route.

**Virtual route (VR):** A virtual route is a bidirectional logical connection between two subarea nodes. At least one end of a virtual route must be in a subarea node that activates virtual routes. All hosts can activate virtual routes.

Eight virtual routes numbered 0 to 7 can be defined between two subarea nodes. One or more virtual routes must be defined for each forward-reverse explicit route pair. A virtual route places a transmission priority on data traffic using the underlying explicit routes.

**Transmission priority (TP):** The transmission priority identifies the priority of message units flowing over an explicit route during a session. The three possible levels of transmission priority are: 0 (lowest), 1, or 2 (highest).

In general, high-priority messages are routed before low-priority messages. Within a specific transmission priority, messages are routed on a first-in, first-out (FIFO) basis.

**Route extension:** A route extension is a logical connection between a subarea node and a peripheral node. A peripheral node uses local addresses for routing

and requires boundary function assistance from an adjacent subarea node to communicate with a nonadjacent subarea node.

Figure 7-3 on page 145 portrays the mapping of a virtual route onto explicit routes. ER0 is the physical path that connects HOSTA to NCPA using transmission group TG1 and NCPA to NCPB using transmission group TG15. The reverse explicit routes (ER0) traverses the same physical path, that is, the same subareas and transmission groups.

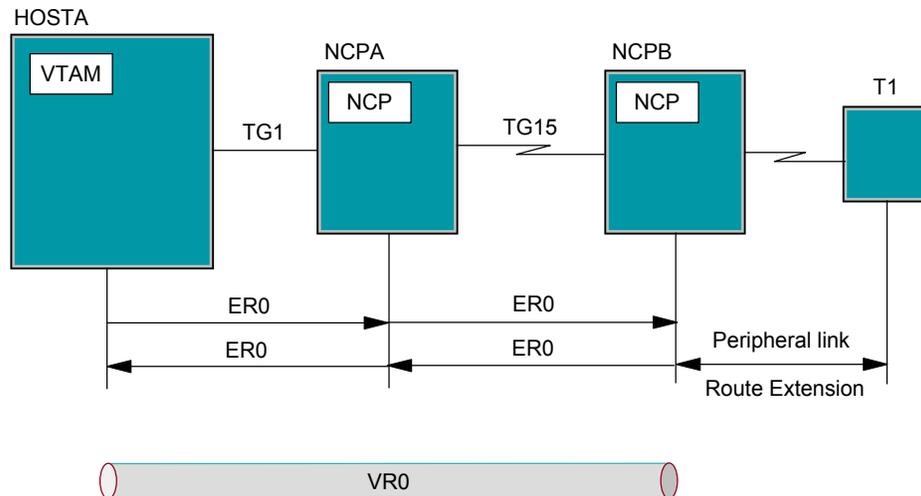


Figure 7-3 Virtual route and explicit route

Virtual route VR0 is the logical connection between the endpoint subareas. VR0 connects HOSTA and NCPB. In our example, VR0 maps onto ER0.

VTAM routing definitions are static and you have to define the routes prior to bringing up the network. While the network is up and running, you can update or add route definitions dynamically.

Figure 7-3 illustrates a very simple and basic subarea network composed of two type 5 nodes. The media connecting the two type 5 nodes is an Ethernet LAN.

HOSTA has been assigned 1 as its subarea number, and HOSTB is defined as subarea 2.

To connect the two hosts, you have to assign and define the explicit route (forward and reverse) and the virtual route. ER0 (explicit route 0) is assigned as the forward and reverse ER. ER0 uses the Ethernet OSA as the physical media.

VR0 (virtual route) is the logical connection between HOSTA and HOSTB. The VR is mapped to ER0 on both sides, HOSTA and HOSTB.

## 7.4 Architectural components of the SNA network

A resource in an SNA network is a *network accessible unit*, which is either an origin or a destination of information transmitted by the transport network (the data link control and path control layers). You already read about control points and system services control points, which are network accessible units. Other network accessible units are:

- ▶ Physical units
- ▶ Logical units

### Physical units

Physical units are components that manage and monitor resources such as attached links and adjacent link stations associated with a node. SSCPs indirectly manage these resources through physical units.

Physical units (PUs) exist in subarea and type 2.0 nodes. (In type 2.1 peripheral nodes, the control point performs the functions of a PU.) The PU supports sessions with control points in type 5 nodes and also interacts with the control point in its own node.

A physical unit provides the following functions:

- ▶ Receives and acts upon requests from the system services control point (SSCP), such as activating and deactivating links to adjacent nodes
- ▶ Manages links and link stations, while accounting for the unique aspects of different link types

### Logical units

End users and applications access SNA networks through *logical units* (LUs), which are the entry point through which users and applications access the SNA network. Logical units manage the exchange of data between end users to applications and application to application, acting as an intermediaries between the two session partners on the two endpoint LUs.

Because SNA is a connection-oriented protocol, prior to transferring data the respective logical units must be connected in a session. In SNA hierarchical networks, logical units require assistance from system services control points (SSCPs), which exist in type 5 nodes, to activate a session with another logical

unit. A session between a logical unit (LU) and an SSCP is called SSCP-LU session. Control information flows from the SSCP to LU session.

A session between two logical units either in the same node or in two different nodes is called an LU-LU session. The session between two LUs is used for application data flows.

All node types can contain logical units. In SNA hierarchical networks, the logical unit has sessions with only one control point in type 5 nodes and with logical units in other nodes. A control point assists in establishing a session between its managed LU and an LU it does not manage in a different node.

Figure 7-4 on page 147 shows LUA, which is managed (owned) by the SSCP in HOSTA and is in session with an application in HOSTB that is not the host that manages the LU.

The control point assists in establishing the session between the two LUs and does not take part in the data transfer between the two LUs.

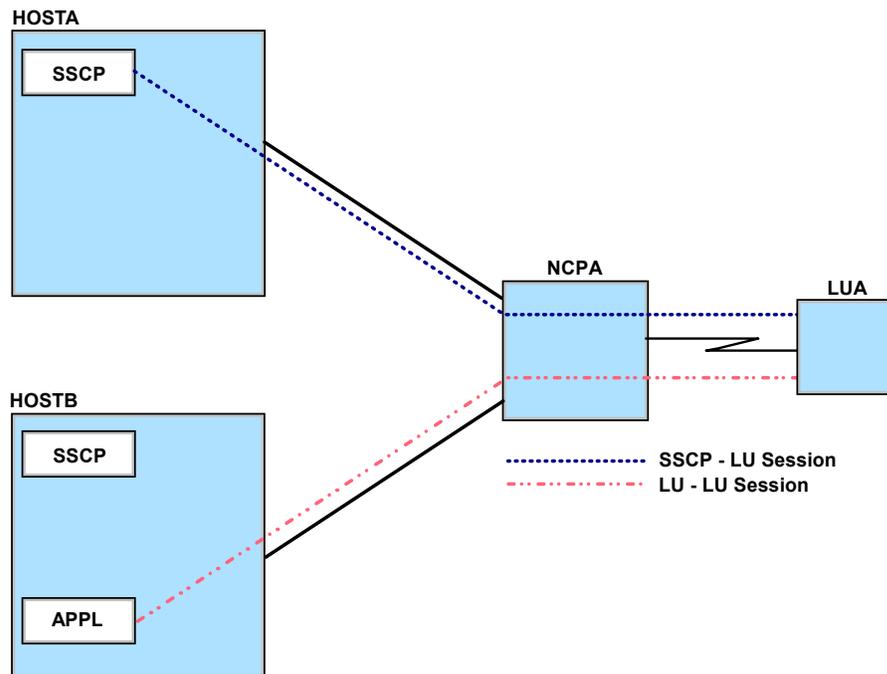


Figure 7-4 LU-LU and SSCP-LU session

## Logical unit types

SNA defines different kinds of logical units called *LU types*. LU types identify sets of SNA functions that support end-user communication. LU-LU sessions can exist only between logical units of the same LU type. For example, an LU type 2 can communicate only with another LU type 2; it cannot communicate with an LU type 3.

The LU types that SNA defines, the kind of configuration or application that each type represents, and the hardware or software products that typically use each type of logical unit are listed here:

- LU type 1** This is for application programs and single-device or multiple-device data processing workstations communicating in an interactive or batch data transfer. An example of the use of LU type 1 is an application program running under IMS/VS and communicating with a 3270 printer.
- LU type 2** This is for application programs and display workstations communicating in an interactive environment using the SNA 3270 data stream. An example of the use of LU type 2 is an application program running under IMS/VS and communicating with an IBM 3270 display station at which an end user is creating and sending data to the application program.
- LU type 3** This is for application programs and printers using the SNA 3270 data stream. An example of the use of LU type 3 is an application program running under CICS/VS and sending data to a 3270 printer.
- LU type 6.2** This is for transaction programs communicating in a client/server data processing environment. The type 6.2 LU supports multiple concurrent sessions. LU 6.2 can be used for communication between two type 5 nodes, a type 5 node and a type 2.1 node, or two type 2.1 nodes.

Examples of the use of LU type 6.2 are:

An application program running under CICS in a z/OS system communicating with another application program running under CICS in another z/OS system.

An application program in a Microsoft Host Integration Server (HIS) or AIX Communications Server communicating with a CICS in a z/OS system.

LU types 1, 2, and 3 are referred to as dependent LUs. An SSCP-dependent LU (or simply dependent LU) requires assistance from a system services control point (SSCP) in order to activate an LU-LU session; therefore, it requires an SSCP-LU session. All non-6.2 LUs are dependent; some type-6.2 LUs are

dependent and some are independent. A type 2.0 node supports only dependent LUs. A type 2.1 node can support any combination of dependent and independent LUs.

LU 6.2 can act either as a dependent or independent LU. An SSCP-independent LU (or simply independent LU) is able to activate an LU-LU session (that is, send a BIND request) without assistance from an SSCP; therefore, it does not have an SSCP-LU session. Only a type-6.2 LU can be an independent LU. A type 2.1 node supports independent-LU protocols to other directly-attached independent LUs in type 2.1 nodes.

### **SNA messages**

In an SNA network, messages flowing through the network contain either a request or a response. Requests are message units that contain:

- ▶ End-user data, called *data requests*. Examples of end-user data include payroll data, personnel data, insurance policy data, and inventory data.
- ▶ Network commands, called *command requests*. Network commands initiate and terminate sessions and control communication between network accessible units.

Responses are message units that acknowledge the receipt of a request. Responses are either positive or negative.

- ▶ Positive responses indicate that a request was received and is acceptable.
- ▶ Negative responses indicate that a request was received, but is unacceptable; they also contain error codes that explain why the request is unacceptable.

### ***Path information unit (PIU)***

In TCP/IP, the unit that is transferred through the network is called a *packet*. A packet includes the data (payload) and the IP and TCP headers. The headers are used to route the packet and manage the TCP session. In SNA the unit that flows in the network is called a *path information unit* (PIU).

As with every networking protocol, messages are routed to an address in the network. In an IP network, the address is the IP address of the host and the address is assigned either dynamically using a DHCP, or it is a static IP address.

In SNA, the messages are sent to a network accessible unit (NAU). The addresses are assigned to the network accessible unit by the control point when the physical units or logical units are activated.

The path information unit carries three fields that are used by the network accessible unit to route the information in the network; see Figure 7-5.

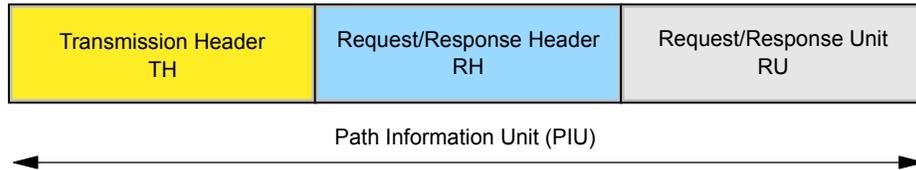


Figure 7-5 Path information unit

### Transmission header (TH)

The transmission header (TH) is used to route message units through the network. The transmission header contains routing information for the transport network. SNA defines different transmission header formats and identifies the different formats by a *form indicator* (FID) type.

Transmission headers vary in length according to their FID type. Path control uses the different FID types to route data between different types of nodes. Two important FID types are:

- ▶ FID2 - This format is used to route data between a subarea boundary node and an adjacent peripheral node, or between adjacent APPN or LEN nodes.
- ▶ FID4 - This format is used to route data between subarea nodes.

Figure 7-6 on page 151 shows the layout of FID2 and FID4 transmission headers and the structure of the addressing field for these two FIDs.

In FID4, the network accessible unit is formed from the subarea and element address. In FID2, used by T2.0 nodes, the element address is 8 bits long, limiting the number of elements to 255. T2.1 also uses FID2, and handles the destination address field (DAF) and origin address field (OAF) as one 16-bit field.

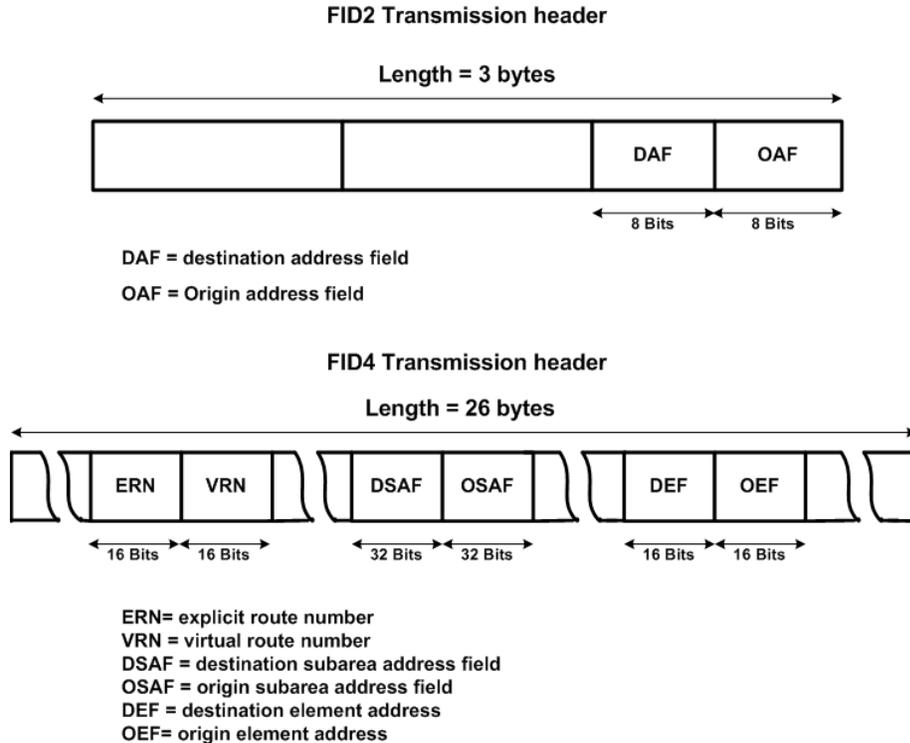


Figure 7-6 SNA addressing with FID2 and FID4 transmission headers

### Request header (RH)

Each request that an NAU sends begins with a request header (RH). A *request header* is a 3-byte field that identifies the type of data in the associated request unit. The request header also provides information about the format of the data and specifies protocols for the session. Only NAUs use request header information.

### Request unit (RU)

Each request that an NAU sends also contains a request unit (RU). A *request unit* is a field of variable length that contains either end-user data (data RUs) or an SNA command (command RUs). Data RUs contain information that is exchanged between end users. Command RUs control the operation of the network.

### Response header (RH)

Each response that an NAU sends includes a response header (RH). Like a request header, a *response header* is a 3-byte field that identifies the type of data

in the associated response unit. A bit called the *request/response indicator* (RRI) distinguishes a response header from a request header.

The receiving NAU indicates whether the response being returned to the request sender is positive or negative by setting a single bit.

### **Response unit (RU)**

A *response unit* (RU) contains information about the request. Positive responses to command requests generally contain a 1-3 byte response unit that identifies the command request. Positive responses to data requests contain response headers, but no response unit. Negative response units are 4-7 bytes long and are always returned with a negative response.

### **Initiating LU-LU sessions**

Because SNA is a connection-oriented protocol, before two entities in the network can communicate, a connection has to be set up between them. End users (application programs and individuals) gain access to an SNA network through logical units and exchange information over LU-LU sessions. Once network resources are active, LU-LU sessions can be initiated.

LU-LU sessions can be initiated in several ways:

- ▶ Either of the participating logical units can initiate an LU-LU session.
- ▶ A network operator can initiate an LU-LU session.
- ▶ A system definition can specify that an LU-LU session be initiated automatically when certain resources become active.

Typically, one of the participating logical units initiates an LU-LU session. The two logical units that communicate with each other over a session are called *session partners*.

Here's how it works:

In SNA, every LU can initiate a session with a partner LU. The LU sends a session request to its control point. The control point locates the partner LU, either within the domain of the control point, or in another domain, or in another network. In an SNA subarea, after the partner LU is located by the SSCP, the virtual route is chosen and a BIND message (RU) is sent by the LU and the session is started. The virtual route is mapped to the explicit route.

LU-LU session initiation generally begins when the session manager in an LU submits a session-initiation request to the appropriate control point. In a subarea network, it can be either the system services control point controlling the LU's domain or, in the case of a type 2.1 peripheral node, the LU's local control point (CP). A session-initiation request specifies the requested session partner's

network name and a mode name. The mode name identifies which set of session parameters that the requesting logical unit chooses for the session. In a subarea network, the mode name is associated with the parameters through a mode table created during system definition.

Using the specified set of session parameters, the control point builds a BIND image. The control point transmits the BIND image in a control initiate request (CINIT request) to the primary logical unit. The primary logical unit (PLU) is the LU responsible for activating the session. The PLU activates the session by sending a bind session request (BIND request, also called a session-activation request) to the secondary logical unit (SLU). The SLU then returns a BIND response to the PLU. A response unit flows between the session partners and the session is started.

The session parameters set up by a BIND include the maximum message size (RU size) that a session partner can send or receive, the protocol used between the session partners, the window size (pacing in SNA terminology), and more.

Figure 7-7 on page 154 illustrates a session initiation using SSCP-dependent protocols.

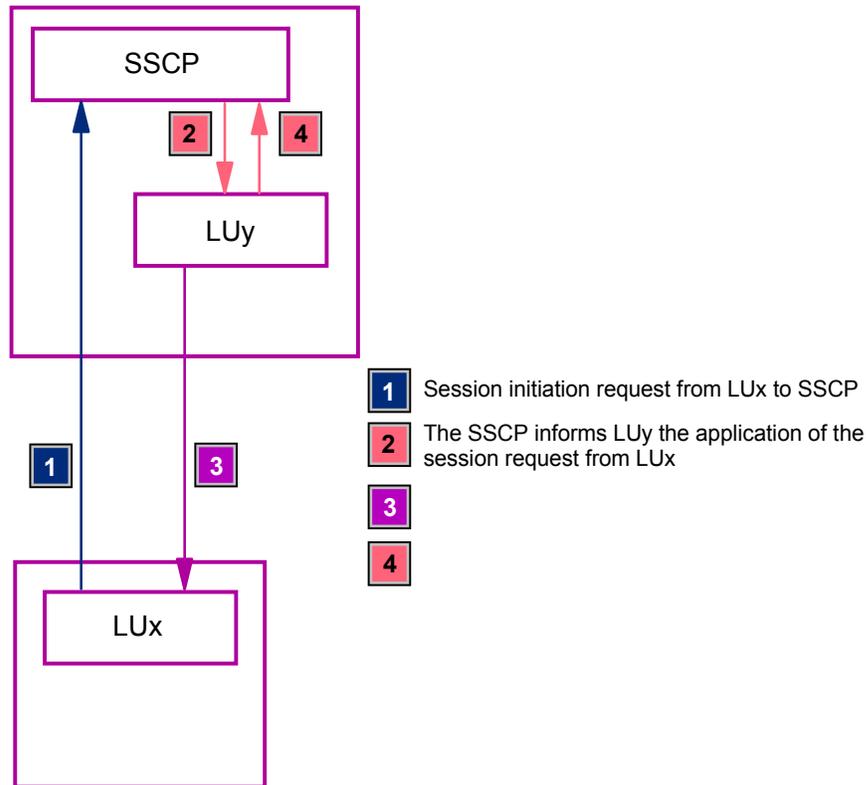


Figure 7-7 Session initiation in SNA subarea network

By comparison, TCP is based on a client-server relationship. The client initiates the connection with the server. The TCP server cannot initiate connections with the client; instead, it listens on a specific port for connection requests. The client must know the IP address of the host where the server exists.

### Class of service (CoS)

In an SNA network, different classes of service can be specified based upon the needs of the end users in the network. SNA's class of service (CoS) is similar to TCP's Type of Service (TOS). The two methods handle the prioritization of messages in the network.

A *class of service* designates the characteristics of a session. It includes such characteristics as security, transmission priority, and bandwidth. The process of defining a class of service is an activity that must take place before a route for LU-LU session is selected.

During session initiation, the class of service for the session is obtained from the session-initiation request, or derived from a mode name specified in the session-initiation request. The route then selected for the session depends on the class of service for the session and available routes.

## 7.5 VTAM subarea definitions

VTAM definitions are located in two data sets referenced by two DDNAME statements in the VTAM started task stored in SYS1.PROCLIB:

- ▶ One data set stores text definitions of the SNA network and is referenced by the VTAMLST DDNAME. The definitions include the various nodes in the network, routing, and hardware components such as channel-to-channel (CTC) connectors and OSA cards.
- ▶ The second data set stores load modules (binary files) and is referenced by the VTAMLIB DDNAME. The files stored are initially coded using VTAM-supplied macros, then assembled and link-edited into the VTAMLIB data set. Examples of these binary definitions include LOGMODE tables and class of service (CoS) tables.

### 7.5.1 Start options

Start options provide information about the conditions under which VTAM runs. They also enable you to tailor VTAM to meet your needs each time VTAM is started. Many options can have defaults specified as start options, thus reducing the amount of coding required. Many start options can be dynamically modified and also displayed. You should be aware that some start options cannot be dynamically modified and require that VTAM be recycled. A complete list of start options is listed in *z/OS Communications Server: SNA Resource Definition Reference*.

To use a start option list, create a data set member named ATCSTR $yy$  and put it in the VTAMLST partitioned data set. The  $yy$  value can be any two characters or numbers and allows you to create different versions of the option list (ATCSTR00, ATCSTR01, ATCSTR02, and so forth) and therefore different versions of VTAM start options.

VTAM is started from the z/OS console or during z/OS system startup with the following command: `START VTAM,LIST= $yy$` . When VTAM initializes, `LIST= $yy$`  determines which option list to use. For example, if you specify `LIST=01`, VTAM uses ATCSTR01. VTAM always first attempts to locate ATCSTR00, regardless of the option list chosen.

If ATCSTR00 does not exist, VTAM sends a warning message to the operator. To avoid receiving this message, create an ATCSTR00 file that contains only comments or start options that are always used for that particular VTAM.

### ***Required start options***

Some start options are required:

<b>SSCPID</b>	The SSCPID start option provides VTAM with a unique numeric identifier. The SSCPID value is used by some physical units to identify the VTAM with which it is in session. If you plan to expand or incorporate a single-domain network into a larger network, be sure that the value of SSCPID is unique for each host. The SSCPID value you specify must also be different from the SSCPIIDs in other networks that can be in session with this host.
<b>SSCPNAME</b>	The SSCPNAME start option provides a unique name for VTAM. This option is required for a single-domain network, but is primarily used in multiple-domain and multiple-network environments to identify a particular VTAM. The SSCPNAME option must be different from the HOSTPU start option that identifies the physical unit within VTAM. <b>Note:</b> The SSCPNAME should match the name that is coded in the cross-domain resource manager major node for this VTAM.
<b>NETID</b>	The NETID start option provides VTAM with the network identifier. If you connect your VTAM to another network, the network identifiers must be unique.
<b>HOSTSA</b>	HOSTSA specifies the subarea number of this VTAM.
<b>HOSTPU</b>	The HOSTPU start option is recommended (but not required) for identifying VTAM to the network. Use the HOSTPU start option to assign a user-defined name to the VTAM host physical unit.

## **7.5.2 Configuration lists**

A *configuration list* specifies the resources that are to be activated when VTAM is started. The names of the resources that are activated when VTAM starts should be placed into an ATCCONxx member in the VTAMLST partitioned data set, where xx is any two alphanumeric characters. The value xx can then be used on the CONFIG operand of the VTAM START command, or on the CONFIG start option in your start option list, to specify which definitions are to be activated at startup. If, as in Table 7-1 on page 160, the configuration list defined is ATCCON01, you can specify CONFIG=01 on the VTAM START command.

Examples of resource definitions that can be included in a configuration list are:

- ▶ Paths
- ▶ Major nodes
- ▶ Minor nodes defined in previously listed major nodes
- ▶ Tables
- ▶ Dynamic reconfiguration files

### 7.5.3 Defining resources to VTAM

In addition to specifying start options and coding configuration lists, you'll need to identify resources in the network to VTAM. Depending on your network, you might need to define a combination of the following resources:

1. Application programs
  - a. Every SNA application program like CICS, IMS, TSO, in-house VTAM applications, and applications developed by other companies, is defined to VTAM.
  - b. When the application initializes, it connects to VTAM and informs VTAM that it is ready to accept requests and service the LUs in the network.
2. Network control programs (NCPs) and peripheral nodes
  - a. The NCP is the software residing in the communication controller. The first task the systems programmer performs is the definition of all the lines, the peripheral equipment (T2.0 and T2.1), links that connect to other SNA domains and the explicit routes that traverse the NCP and virtual routes that originate in the NCP.
  - b. During VTAM initialization, VTAM connects to the NCP, activates the resources in the NCP, and determines if a load (IPL) of the NCP is required.

3. Physical units (PUs) and logical units (LUs)

PUs and LUs are defined either in the NCP or in VTAM. PUs are connected to the mainframe by any of the following means:

- a. Directly attached through a channel (either parallel or ESCON)
- b. Leased lines to the NCP
- c. Switched (dialed) lines to the NCP
- d. LAN-connected, either directly to an NCP attached LAN, or directly to the mainframe using an OSA card

PUs connected through dialed lines and LAN-connected PUs are defined to VTAM as switched major nodes. The definition includes the PU and the LUs associated with that PU. Because a switched connection is casual (comparable to a phone call) and un-authorized people can dial the

mainframe or connect to a LAN jack, the switched major node includes some very basic security definitions that can be used to identify the switched major node during the connection phase.

4. If your network is a multiple domain network, additional definitions are required. These definitions include:
  - a. Cross-domain resource manager (CDRM)

CDRM defines the logical connection to an adjacent mainframe. The CDRM connection is used to send and receive control information between the two mainframes.
  - b. Cross-domain resource (CDRSC)

CDRSC is an SNA resource that exists in the domain of another host and sometimes sets up or actually maintains a session with a local resource in the domain of the host where the CDRSC is defined.
  - c. Adjacent SSCP table (ADJSSCP)

The ADJSSCP is used to control the search for SNA resources in a multi-domain SNA network. The ADJSSCP determines the order the search is performed, or in other words, which mainframe is queried first to determine if it owns the resource, and the subsequent hosts if the query was not successful.

Any of these resources can be predefined to VTAM using a static definition statement. For example, applications are predefined using an application major node and application (APPL) statements; switched PUs are predefined using a switched major node for the PUs and LUs. In addition, many VTAM resources can be dynamically defined as VTAM learns of them. A switched PU may be defined when it dials in. Not all resources need be initially defined to subarea nodes. There are several methods by which resources may be dynamically defined to the network.

Although static and pre-defined resources are burdensome, many mainframe installations are reluctant to allow dynamic definition of PUs or LUs. The rationale for preventing dynamic definition is security. When every resource is pre-defined the installation has more control over who accesses the mainframe.

Another method for dynamic, but controlled, definition is called dynamic reconfiguration, which allows a system programmer to add or delete peripheral nodes dynamically. The resources are added or deleted through a VTAM major node configuration statement (stored as a member in SYS1.VTAMLST) or by using VTAM commands entered from the console.

Following are examples of dynamic definitions of SNA resources and how VTAM determines the credentials and attributes for dynamically defined resources.

Dynamic definition of independent LUs allows an independent LU to activate an LU-LU session with an LU in a subarea network without prior definition of the independent LU to the subarea network. The SSCP controlling the independent LU's owning node dynamically stores the name and location of the independent LU as determined from the bind request sent by the independent LU. Subsequently, other LUs in the network can activate LU-LU sessions with the independent LU, because the independent LU is then known to the subarea network.

Dynamic definition of dependent LUs allows dependent LUs to be defined using information that specifies how many dependent LUs can be on switched or non-switched lines.

Dynamic definition of switched resources uses information from the exchange ID (XID) message exchange for switched PUs to allow switched PUs and their associated dependent LUs to be dynamically defined. As with dynamically-defined independent LUs, other LUs in the network can activate LU-LU sessions with the dynamically defined dependent LU, because the dependent LU is then known to the network.

Dynamic PU definition dynamically defines an adjacent link station (or PU). This method, like dynamic definition of switched resources, relies on information obtained from the XID to create dynamically a definition of the resource. After the adjacent link station is defined, it may be used for connectivity to its independent logical units.

### ***Sift-down effect***

The *sift-down effect* enables you to code an operand on a higher-level node so that you do not need to recode it on each lower-level node for which you want the same value. As a result, the sift-down effect greatly simplifies the coding process. *z/OS Communications Server: SNA Resource Definition Reference* identifies and describes the definition statements and operands to which sifting applies.

## **7.5.4 Sample network**

The network in Figure 7-8 on page 160 demonstrates how to define this subarea network. The network consists of two mainframe hosts named HOSTA and HOSTB. The two mainframes are connected using Ethernet LAN implemented by an OSA in every host.

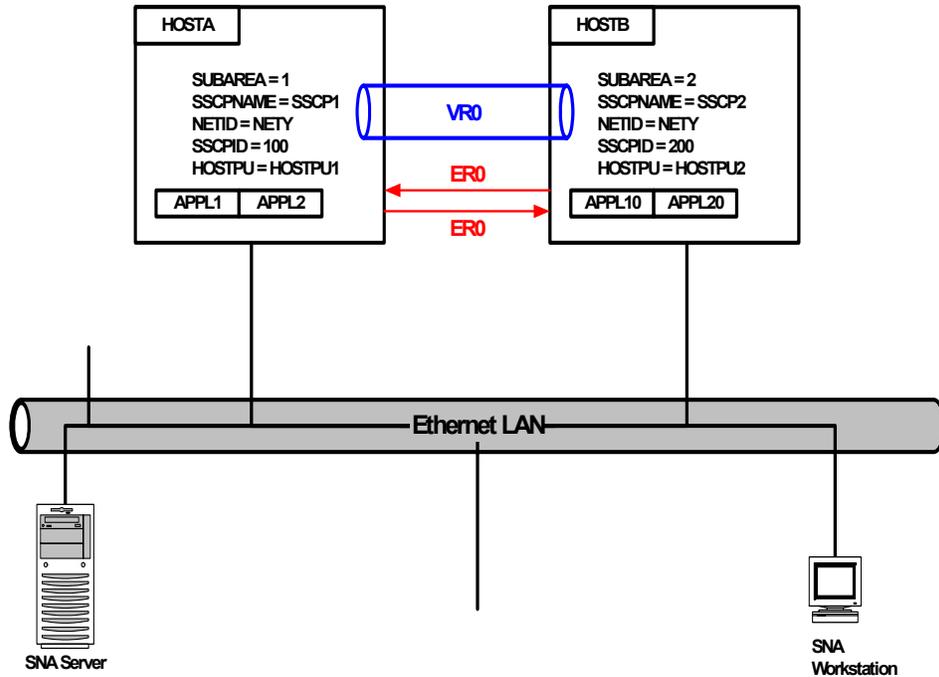


Figure 7-8 Sample network

The sample network in Figure 7-8 has matching configuration definitions in Table 7-1. The definitions on the left side of the table are for HOSTA, and those on the right side relate to HOSTB.

Table 7-1 Configuration definitions matching the sample network

HOSTA	HOSTB
<b>ATCSTR01 Definition</b> HOSTSA=01, SSCPNAME=SSCP1, NETID=NETY, SSCIP=100, HOSTPU=HOSTPU1, CONFIG=01	<b>ATCSTR02 Definition</b> HOSTSA=02, SSCPNAME=SSCP2, NETID=NETY, SSCIP=200, HOSTPU=HOSTPU2, CONFIG=02
<b>PATH Definition</b> PATH01 PATH DESTSA=2, ERO=(2,1), VRO=0	<b>PATH Definition</b> PATH01 PATH DESTSA=2, ERO=(2,1), VRO=0

HOSTA	HOSTB
<p><b>XCA (OSA) major node for attaching peripheral and subarea nodes</b></p> <p>OSA1 VBUILD TYPE=XCA  PORT1 PORT ADAPNO=1,  CUADDR=29A,  MEDIUM=CSMACD</p> <p>GRP1A GROUP DIAL=YES,  AUTOGEN=(5,L,P),  CALL=INOUT,  <b>ANSWER=ON</b></p> <p><b>GRP1B</b> GROUP DIAL=NO  LN1B LINE  PU1B PU SUBAREA=2,  PUTYPE=5,  TGN=1,  MACADDR=note 1</p>	<p><b>XCA (OSA) major node for attaching peripheral and subarea nodes</b></p> <p>OSA2 VBUILD TYPE=XCA  PORT2 PORT ADAPNO=1,  CUADDR=83C,  MEDIUM=CSMACD</p> <p>GRP2A GROUP DIAL=YES,  AUTOGEN=(5,L,P),  CALL=INOUT,  <b>ANSWER=ON</b></p> <p><b>GRP2B</b> GROUP DIAL=NO  LN2B LINE  PU2B PU SUBAREA=1,  PUTYPE=5,  TGN=1,  MACADDR=note 1</p>
<p><b>CDRM major node</b></p> <p>CDRM1 VBUILD TYPE=CDRM  NET1 NETWORK NETID=NETY  SSCP1 CDRM SUBAREA=1  CDRM2 CDRM SUBAREA=2</p>	<p><b>CDRM major node</b></p> <p>CDRM2 VBUILD TYPE=CDRM  NET2 NETWORK NETID=NETY  SSCP2 CDRM SUBAREA=2  CDRM1 CDRM SUBAREA=1</p>
<p><b>Application major node</b></p> <p>APPL1X VBUILD TYPE=APPL  APPL1 APPL ACBNAME=APPL1  APPL2 APPL ACBNAME=APPL2</p>	<p><b>Application major node</b></p> <p>APPL2X VBUILD TYPE=APPL  APPL10 APPL ACBNAME=APPL10  APPL20 APPL ACBNAME=APPL20</p>
<p><b>Switched PU major node (SNA server)</b></p> <p>SWPU1 VBUILD TYPE=SWNET  PU1 PU MAXDATA=1033,  ADDR=01,  CPNAME=SNASRV, R,  PUTYPE=2</p>	<p><b>Switched PU major node (SNA workstation) with dependent LUs</b></p> <p>SWPU2 VBUILD TYPE=SWNET  PU2 PU MAXDATA=1033,  ADDR=01,  IDBLK=01A,  IDNUM=,30D54,  PUTYPE=2</p> <p>LU1 LU LOCADDR=1  LU2 LU LOCADDR=2</p>
<p><b>CDRSC major node (Independent LU 6.2)</b></p> <p>CDRSC1 VBUILD TYPE=SWNET  ILU1 CDRSC ALSLIST=PU1</p>	

HOSTA	HOSTB
<b>ATCCON01</b> PATH01, OSA1, CDRM1 APPL1X, SWPU1, CDRSC1	<b>ATCCON02</b> PATH02, OSA2, CDRM2 APPL2X, SWPU2, CDRSC2

## 7.6 Summary

An SNA network implements the IBM proprietary networking architecture. Subarea networking carries out a hierarchical network paradigm. The information about the network resources and the definitions are stored in the mainframe that is located at the top of the hierarchy. The hierarchical structure of the network follows the organizational structure of many enterprises and that's one of the reasons that SNA was accepted and implemented by many enterprises.

In an SNA subarea network, the routes between subarea nodes and almost all the resources are pre-defined.

In TCP/IP, the unit that is transferred through the network is called a packet. A packet includes the data (payload) and the IP and TCP headers used to route the packet and manage the TCP session. In SNA the unit that flows in the network is called a path information unit (PIU). It carries the data, the SNA headers, the transmission header (TH), and request header (RH).

SNA in general has robust data flow control that permits constant flow of data to and from the network and prevents congestion in network resources.

Although today TCP/IP is the dominant networking protocol in almost all organizations, SNA applications will exist for the foreseeable future due to the massive investment made in SNA applications.

Key terms in this chapter		
APPN network	ATCSTRxx	explicit route
hierarchical network	logical unit (LU)	network control program (NCP)
node type	peer network	physical unit (PU)
session	subarea network	switched major node

Systems Network Architecture (SNA)	systems services control point (SSCP)	transmission group
virtual route	Virtual Telecommunications Access Method (VTAM)	





# SNA Advanced Peer-to-Peer Networking (APPN)

**Objective:** After completing this chapter, you will be able to:

- ▶ Define APPN.
- ▶ Describe the APPN node types.
- ▶ Explain how APPN routes SNA data.
- ▶ Explain the role of an APPN control point.
- ▶ Describe an extended border node.
- ▶ Define high performance routing.
- ▶ Summarize APPN support for dependent LUs.
- ▶ Explain how to define VTAM APPN networks.
- ▶ Explain how to define a combined VTAM and APPN network.

## 8.1 Introduction

In the mid-1980s, SNA subarea networking was the dominant networking protocol used for data processing. Its robustness, management tools, and

predictable response time attracted many organizations to SNA, and they used SNA in their mission-critical applications.

The major drawback of SNA subarea network was the requirement to provide static definitions for most SNA resources.

At the same time, intelligent workstations were proliferating. The hierarchical nature of subarea SNA was not suitable for these workstations, which required peer connections and dynamic definitions.

Another criticism users of SNA subarea networking had addressed session continuity. Although subarea networks use alternate routes, failure of hardware or software components along the route causes the sessions along the route to fail. Although sessions can be reestablished over alternate routes, the process affects the end-user's session availability.

IBM developed APPN to reduce to a minimum the task of defining SNA resources and routes. The definitions are limited to the local APPN node where one defines the name assigned to the resources, the attachment (LAN or WAN) to be used, and the node type.

APPN learns the network topology and the location of the various nodes in the local network, and searches for resources in the network and adjacent networks. When establishing a session, APPN selects the best available route between the session partners.

Some APPN nodes implement intermediate session routing. Nodes that support intermediate session routing are used along the session path to route session data between the two session endpoints.

The initial APPN implementation did not address the session continuity problem, and when organizations started to implement APPN they realized that the performance of the intermediate session routing function was poor. IBM went "back to the drawing board" and developed an extension to APPN, called high performance routing (HPR).

HPR introduced the rapid transport protocol (RTP) and automatic network routing (ANR). These two added functions address session continuity and the performance issue of intermediate session routing. With HPR, a session is switched to an available route without disrupting the session. The end user is not aware that a failure took place along the path of the session.

## 8.1.1 Advanced Program-to Program Communications (APPC)

APPC, also known as LU 6.2, was introduced by IBM in 1982 to address the exchange of data between two peer programs that are located either in the same computer or in two systems connected by the network.

APPC is an architecture that defines a set of networking protocols and an application programming interface (API). Because APPC is a networking protocol, it does not address the individual programming language syntax.

The APPC API is described as an abstract presentation of the various API functions called *verbs*. The verbs define the sequence and order an application program has to follow in order to communicate with peer application programs. The implementation of APPC in the various programming language converts the abstract API to functions or callable subroutines that conform to the syntax of the programming language.

To avoid the hierarchical nature of the mainframe SNA, and allow “small” computers like programmable workstations and iSeries™ to use APPC, the type 2.1 node (T2.1) was introduced. A type 2.1 (T2.1) node is a peer-oriented peripheral node that attaches to a mainframe, a communication controller, or another peripheral node. For a refresher on basic node types, see 7.3.1, “SNA nodes” on page 138.

APPN was designed to support APPC in T2.1 nodes and the exchange of control information between two APPN nodes uses APPC protocol.

## 8.1.2 Advanced Peer-to-Peer Networking (APPN)

An Advanced Peer-to-Peer Networking (APPN) network is composed of a group or groups of connected T2.1 nodes. T2.1 nodes provide sessions between LUs and peer-level connectivity using the APPC protocol. Unlike hierarchical SNA subarea networks, sessions between two APPN or low-entry networking nodes can be established without involving a mainframe in the session setup.

APPN nodes:

- ▶ Search and keep track of SNA resource locations within the network
- ▶ Dynamically exchange information about the resource location they own, eliminating the need to pre-define resources owned by other nodes
- ▶ Maintain knowledge of APPN network topology (the APPN nodes and the links between them) and use this information to select the best available path to route sessions between SNA resources, thereby eliminating the need for complex path definitions

### 8.1.3 APPC versus APPN

As their names imply, APPC, or Advanced Program-to-Program Communication, deals with programs, while APPN, or Advanced Peer-to-Peer Networking, deals with networks. APPC defines the rules of how programs exchange information.

These rules do not deal with the details of network setup and routing. It is APPN that defines how APPC traffic gets from one point to another in a network. A reasonable comparison between APPC and APPN is the difference between a person using the telephone and the services the telephone company offers.

#### **APPC**

For example, when you want to call someone, you look up the telephone number and then enter it. Both parties identify themselves and the exchange of information begins. When the conversation is finished, both parties say good bye and hang up. This protocol, although informal, is generally accepted and makes it much easier to communicate.

APPC provides the same functions and rules, only between application programs instead of people. An application program tells APPC with whom it needs a conversation. APPC starts a conversation between the programs so they can exchange data. When all the data has been exchanged, APPC provides a way for the programs to end the conversation.

#### **APPN**

APPN provides networking functions similar to those provided by the telephone companies. After dialing a telephone number, the telephone network routes the call through trunks, switches, branches, and so on.

To make the connection, the network takes into consideration what it knows about available routes and current problems. This happens without the caller understanding the details of the network. A person is able to talk on the telephone to another person no matter where they are or no matter how the call was routed.

APPN provides these functions for APPC applications and their data. It computes routes for APPC communication through the network, dynamically calculating which route is best. Like the telephone company, APPN's routing is done transparently. APPC applications cannot tell whether the communications partner in the APPN network is located in the same computer, one office away, or in another country. Similarly, if someone moves within the same city and takes their phone number, the phone network handles the change with no other user impact.

## 8.2 APPN node types

The building block for an APPN node is the T2.1 (low-entry networking, LEN) node. Be aware that T2.1 by itself does not provide any APPN functionality; additional software is required to make a T2.1 node an APPN node.

Depending on the software that implements APPN in T2.1 nodes, the node can be configured in the APPN networks with varying complexity, from the simplest case of an isolated pair of low-entry networking nodes to a large APPN network. Using low-entry networking or APPN protocols, any node can control the establishment and termination of sessions.

The following node types can be implemented by T2.1 nodes:

- ▶ Low-entry networking (LEN)
- ▶ APPN end node (EN)
- ▶ APPN network node (NN)

To ease the migration from subarea networking to APPN in the mainframe, the following nodes types can be implemented in a mainframe:

- ▶ Interchange node (ICN)
- ▶ Migration node

T5 and T4 nodes also support low-entry networking and APPN protocols, and are fully compatible with T2.1 nodes in these contexts. They also introduce product features of their own, related to enhanced subarea-APPN interchanges. When a subarea node implements either APPN or low-entry networking protocols, it acts as a T2.1 node and can still implement, depending on VTAM's definitions, the subarea T5 and T4 functions.

### 8.2.1 Low-entry networking (LEN) nodes

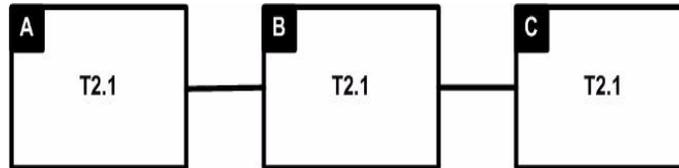
Low-entry networking nodes, also referred to as T2.1 nodes, were introduced in the mid-1980s to address the requirement for peer networking. The low-entry networking node was the first stage of the APPN evolution.

The T2.1 node allows peer-to-peer connection and provides the physical and session-level connectivity required to support logical unit type 6.2 (LU 6.2). T2.1 nodes use protocols with reduced system definition requirements. For example, link station roles are negotiated as primary or secondary during the connection phase, instead of, as in the case of subarea networking, being predefined.

Low-entry networking nodes do not implement a control point. With low-entry networking, you must predefine every partner resource (if it does not reside on this same node) along with the first hop (link) toward that resource (this is

because neither searching nor topology exchanges are supported by low-entry networking nodes). This predefinition requirement is the primary drawback of low-entry networking nodes.

Figure 8-1 illustrates a T2.1 skeleton network. The lines connecting the three nodes represent networking infrastructure like LAN, serial lines, or frame relay ATM. The T2.1 nodes are low-entry networking nodes without any APPN functionality.



*Figure 8-1 T2.1 peer connectivity*

In the T2.1 connection illustrated in Figure 8-1, LU-LU sessions can be established between nodes A and B and nodes B and C. Because node B does not include a control point or any APPN functions, it cannot route sessions, and therefore node A and node C cannot establish LU-LU sessions.

Prior to APPN, when only T2.1 nodes were implemented in SNA, transferring data from node A to node C required either a direct physical link between node A and node C or an application program in node B that relays messages between nodes A and C.

## 8.2.2 End nodes (EN)

APPN end nodes implement a subset of the full APPN functions and rely on one of the network nodes they are attached to for accessing the network, locating resources, and providing routing services.

- ▶ All end nodes provide a peer environment for LU-LU (independent LUs) sessions where one LU resides in the local end node.
- ▶ The end node is limited in what it can do on its own. It requires the cooperation from an adjacent network node server with which the end node has established a CP-CP sessions. The adjacent network node server assists the end node in locating session partners, choosing session paths, and routing the bind to establish sessions.
- ▶ APPN end nodes also can register their local LUs and local topology (links to other network nodes and end nodes) to their network node server as well, thereby eliminating the need to send these EN transmission group vectors (TGVs) on every APPN locate request/reply. By reducing the number of

searches sent to an EN, the EN can dedicate more of its resources (CPU, memory, and so on) to more productive work.

- ▶ End nodes do not participate in network topology exchange, but they do maintain their local topology.
- ▶ An APPN end node can have links to multiple nodes, including multiple network nodes, but may have CP-CP sessions with only one network node at a time (its network node server).
- ▶ Multiple attachment points between an end node and the APPN network may be desirable for increased throughput and high network availability. Attachment to multiple network nodes allows the end node to switch to a different network node server if its original network node server fails or connectivity to it is lost.
- ▶ An APPN end node can attach to any low-entry networking or APPN node.
- ▶ An end node is not required to have the same NETID as its network node server or any adjacent node.

### 8.2.3 Network nodes (NN)

An APPN network node implements full APPN functionality and services. A collection of connected network nodes comprises the APPN backbone. Figure 8-2 illustrates an APPN backbone with four network nodes, NN1 through NN4.

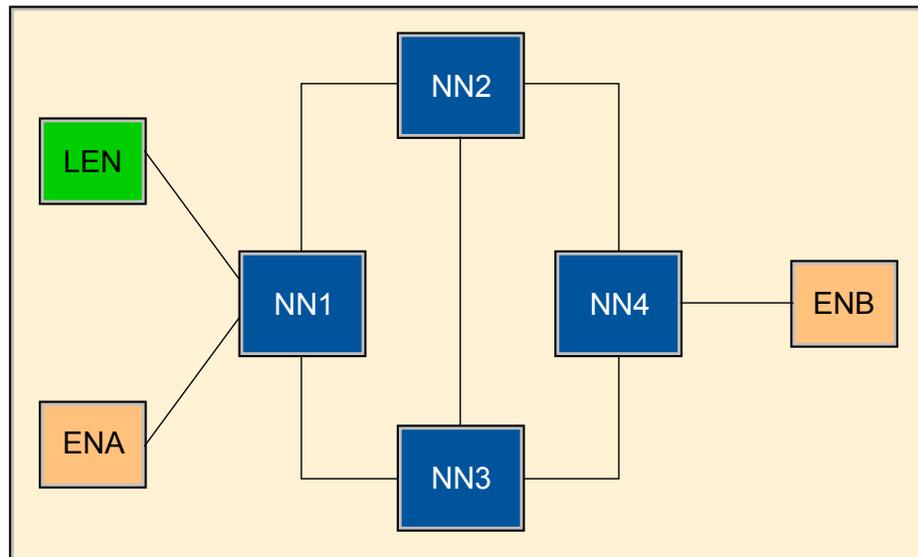


Figure 8-2 APPN backbone

- ▶ An APPN network node manages and supports its own resources (LUs defined in the network node) and that of its served APPN end nodes and low entry networking nodes.

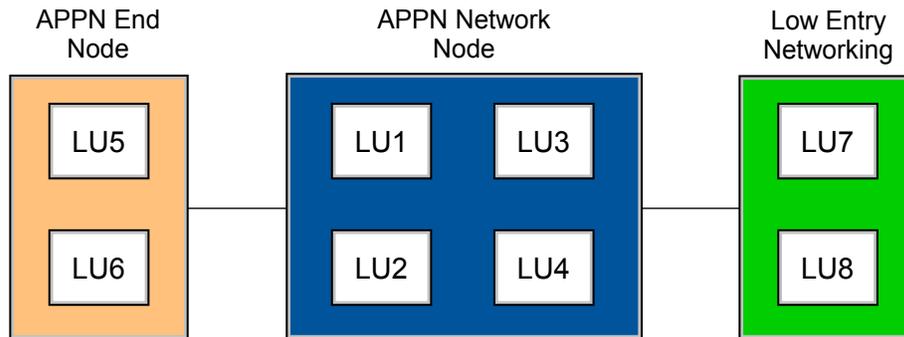


Figure 8-3 Network node-owned resources LU1 through LU4

In Figure 8-3 on page 172, LU1 through LU4 are locally-owned resources in the APPN network node. LU5 through LU8 reside in different T2.1 nodes. Only LU5 and LU6 in the APPN end node are serviced by, and known to, the APPN network node.

To make the network node aware of the existence of the LU7 and LU8, these LUs must be defined manually to the network node because they are owned by a low-entry networking node. If any LU residing in the low-entry networking node must establish a session to any of these LUs, LU1 through LU6 must be predefined on the low-entry networking node.

- ▶ An APPN network performs intermediate routing of data on sessions that traverse it.

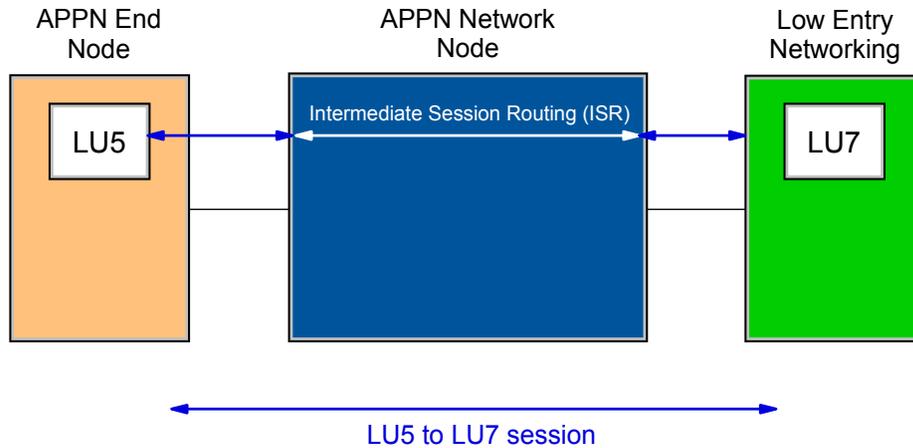


Figure 8-4 APPN intermediate session routing (ISR)

Figure 8-4 is an example of how session routing is implemented in APPN. The diagram illustrates how the session path between LU7 in the low-entry networking node and LU5 in the end node traverses the network node and eventually is routed through the network node. The APPN network node performs intermediate session routing (ISR) for the data transferred on the LU-LU session between LU5 and LU7.

- ▶ An APPN network node provides network searches, network topology management, session route selection, and services to its own LUs and end nodes attached to it.
- ▶ An APPN network node can be a session endpoint (one of its own LUs could be a session partner to an LU in another APPN node) or an intermediate node on a session path.

One deficiency of APPN is that searches for resources can flood the network and eat up too much bandwidth. A network node server is one of the functions that can reduce the searches in the network. The term network node server refers to a network node's role in providing network services for specific APPN end nodes attached to it.

## 8.2.4 Specialized network node types

APPN was designed for implementation on a wide range of hardware platforms and operating systems, including programmable workstations, desktop computers, UNIX and Windows servers, and the IBM mainframe.

The APPN function and services for the various platforms are different, depending on the APPN node role. Therefore, APPN architecture defines basic and optional sets. All APPN nodes must adhere to the basic set of functions according to their node type (network node or end node) and they can implement one or more option sets. The requirement that an APPN node must implement the basic set defined for its role (network node or end node) assures that the node can establish APPN sessions with its peer node.

A node that implements the basic set can communicate with nodes that implement additional option sets. The two nodes learn the optional capabilities of other nodes in the network when they connect to each other and/or when they exchange network topology. (Network nodes can learn about optional capabilities of other non-adjacent network nodes through the topology database.) If an optional set is implemented in a node, the complete optional set should be implemented. There is no ability to implement subsets an optional set.

The following specialized network node types are examples of optional sets of functions.

### **Central directory server (CDS)**

A central directory server (CDS) is implemented only in a network node. It provides more extensive functionality than the directory services in a basic network node.

When a network node receives a search request, it checks its database for the resource. If it does not find the resource in its database, it sends the request to a central directory server if one exists in the network.

When the central directory server receives a search request, it checks its database for the resource. If the central directory server that received a search request locates the resource in its own database, it verifies the information and sends a reply to the originating network node server with the location of the requested LU.

If it does not find the resource in its database and there are other central directory servers in the network, it sends the search to the other central directory servers only. If the central directory server receives a positive reply from any of the other central directory servers, it verifies the information, updates its own database with the information, and notifies the originating network node of the location of the requested LU.

If the central directory server receives negative replies from all the other central directory servers (or if there are no other central directory servers in the network), it initiates a broadcast search. The search steps performed in an APPN network are explained in 8.5.2, "Locating resources" on page 187.

The network nodes learn about the existence and location of the central directory server through the topology database. At any given point in time, every network node knows where every reachable central directory server exists in the network.

### Extended border node (EBN)

Independent SNA networks might have a requirement to be interconnected. For instance, mergers and acquisitions might require interconnecting two SNA networks or two business partners might need to exchange information.

Two subarea networks can be interconnected through SNA network interconnection (SNI). SNI is an SNA-defined architecture that enables independent subarea networks to be interconnected through a gateway.

An extended border node (EBN) is a network node capable of multiple APPN network connections, and it can maintain CP-CP connectivity with a network node that has a different NETID. Figure 8-5 on page 175 illustrates two SNA networks connected using extended border nodes.

APPN topology information does not cross the extended border node connection or APPN subnetwork boundary, but search requests can, and an LU-LU session can be set up. An APPN subnetwork boundary is assumed when an extended border node is connected to a network node (or extended border node) with a different network identifier.

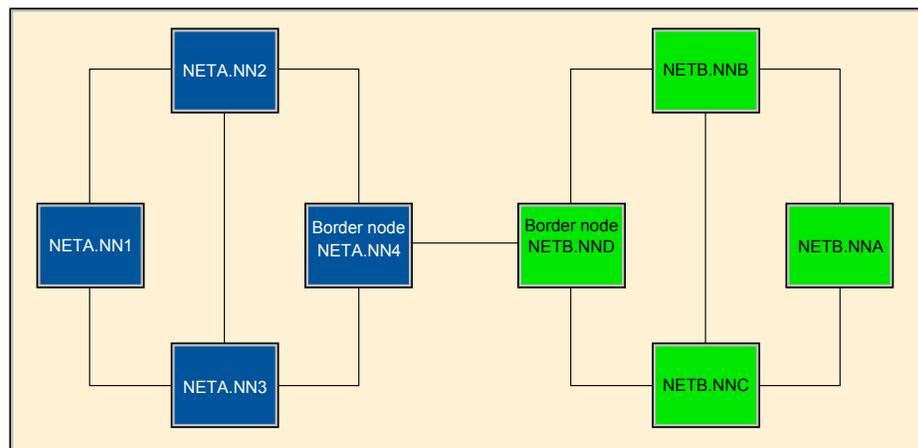


Figure 8-5 Two SNA networks connected using extended border nodes

### Branch extender (BEX or BrEx or BrNN)

Branch extender is an extension to the APPN architecture that allows an APPN node to appear as a network node to the downstream end nodes and low-entry networking nodes and as an end node to the wide area network (WAN); see

Figure 8-6. Implementing branch extender eliminates topology and APPN broadcast search flows between the WAN (mainframes) and the branch office.

The operations staff of the mainframe is not interested in whether a workstation in the branch is booted or powered off. The branch extender isolates the mainframe from the networking equipment in the branch. The topology and directory server of the network node part of the branch extender store the information about the branch networking equipment. The information is not propagated to the mainframe APPN databases.

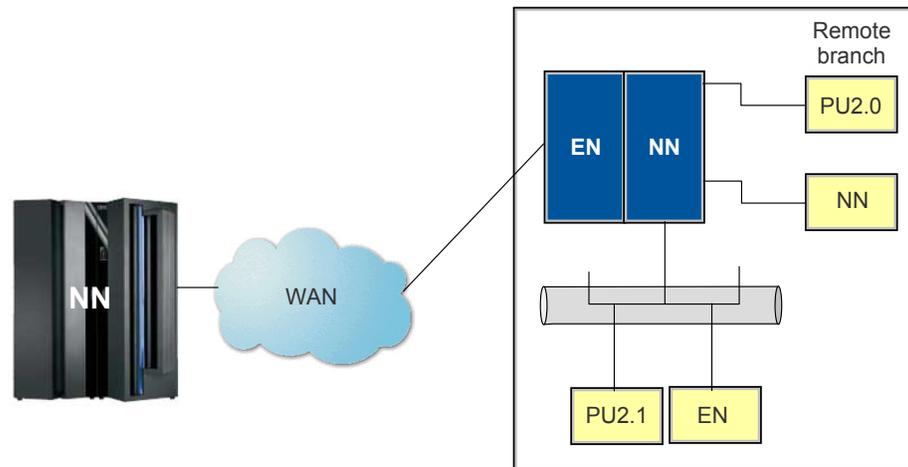


Figure 8-6 Example of a branch extender

## 8.2.5 Specialized VTAM Nodes

As the name implies, “the specialized VTAM nodes” are implemented only in the mainframe. These nodes enable the mainframe to connect directly to both subarea and APPN networks.

### Interchange (network) nodes (ICNs)

An interchange node resides on the border of an APPN network and a subarea network. It provides protocol conversion between subarea and APPN networks to enable the integration of the two types of networks. Because an interchange node can convert session requests from one protocol to the other and can provide intermediate routing, it can establish sessions from one type of network to the other.

An interchange node combines the function of a subarea node and a network node. It controls resources and functions as a network node in the APPN network and as an SSCP and a cross-domain resource manager (CDRM) in the

subarea network. All of the characteristics described for network nodes and subarea nodes apply to interchange nodes.

An interchange node:

- ▶ Uses subarea path definitions to determine routes within the subarea network
- ▶ Uses the topology database to determine routes within APPN networks
- ▶ Uses both SSCP-SSCP and CP-CP sessions to communicate with other nodes
- ▶ Has a subarea number and is defined as a network node, NODETYPE=NN
- ▶ Can own and activate network control programs (NCPs)

The interchange node communicates network control data by using SSCP-SSCP sessions with other subarea nodes and CP-CP sessions with other APPN nodes. To enable it to participate in the subarea network, it is defined with a unique subarea number and requires subarea path definition statements. It can be connected to other APPN nodes, low-entry networking nodes, and subarea nodes.

Many IBM mainframe installations implement interchange nodes because both APPN and subarea components must co-exist in most mainframe networks. Those installations still have subarea networking but are starting to implement APPN.

VTAM determines the node type of the mainframe using two parameters in VTAM's start option. If HOSTSA is set to a subarea number and NODETYPE=NN, an interchange node is implemented by VTAM. If HOSTSA is not specified and CONNNTYPE=NN or EN, VTAM implements an APPN node.

### **Composite network nodes (CNNs)**

Because a Network Control Program (NCP) does not have a control point, NCPs cannot function as APPN nodes by themselves. Instead, NCPs work with their owning VTAM (the one that activated the NCP) to present the appearance of single APPN node to other APPN nodes. This collection of a VTAM network nodes and its owned NCPs is referred to as a composite network node; note the following:

- ▶ The composite network node can have either APPN functions only, or both APPN and subarea functions.
- ▶ Existing subarea protocols are used within the composite network node for communication between the T5 node and its T4 nodes.

- ▶ APPN protocols are used to communicate with other APPN network nodes and end nodes. The T4 node provides boundary function services for attaching other APPN nodes.

The rationale for composite network nodes is to ease the migration from subarea network to APPN. With composite network nodes, an installation can preserve its current hardware while still providing a migration path to APPN networking. Figure 8-7 on page 178 shows a composite network node.

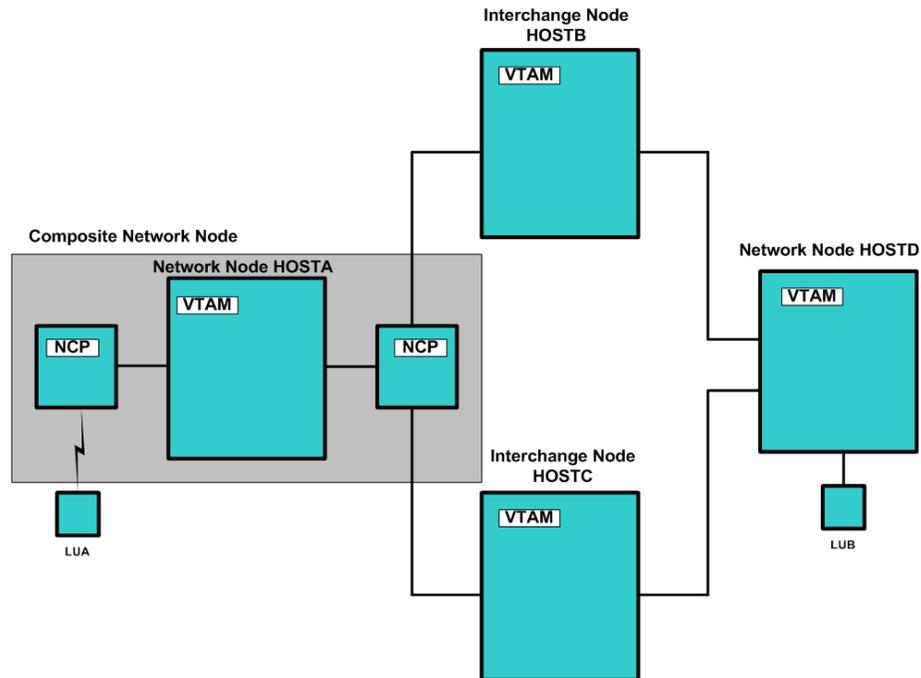


Figure 8-7 Composite network node

A composite network node configuration provides the functional combination of a single T5 (VTAM) node. All the T4 (NCP) nodes that the composite network node owns appear as one logical APPN network node to other low-entry networking and APPN nodes to which it is interconnected.

Figure 8-7 shows a composite network node connected to two VTAM hosts (HOSTB and HOSTC) acting as an interchange node. The interchange node supports SSCP-SSCP sessions with other VTAM nodes as well as CP-CP sessions with adjacent APPN network nodes and end nodes. This enables the interchange node to use both APPN and subarea data flows to locate LUs. From the APPN node's viewpoint, LUs owned by subarea VTAMs (for example, LUB on HOSTD) appear to reside on APPN end nodes.

## Migration data hosts (MDHs)

A migration data host (MDH) combines the function and roles of an APPN end node and a subarea node, and resides on the periphery of a combined APPN and subarea network. A migration data host:

- ▶ Uses subarea network routing definitions
- ▶ Does not perform intermediate session routing or interchange node functions in combined APPN/subarea network.
- ▶ Uses CP-CP and SSCP-SSCP sessions to communicate with other nodes
- ▶ Is defined as an end node, NODETYPE=EN
- ▶ Can attach to NCPs over APPN or subarea links, but cannot activate NCPs
- ▶ Has a subarea number defined on the HOSTSA start option

Like a data host in a subarea network, a migration data host is dedicated to processing application programs and does not control network resources. It also participates as a cross-domain resource manager (CDRM) in the subarea network. The migration data host also functions as an end node in the APPN network. All of the characteristics previously described for end nodes apply to migration data hosts.

To enable the migration data host to participate in the subarea network, it is defined with a unique subarea number and supports subarea path definition statements.

The migration data host communicates network control data by using SSCP-SSCP sessions with other subarea nodes and CP-CP sessions with its network node server. It can be connected to other APPN nodes, low-entry networking nodes, and subarea nodes.

## 8.3 Control point (CP-CP) sessions

To perform directory services and topology and route selection services, adjacent APPN nodes throughout the APPN network use the pair of CP-CP sessions to exchange network and control information.

CP-CP sessions are always logical unit type 6.2 (LU 6.2) sessions. Using this session type, a contention situation could arise if both session partners attempted to allocate a conversation and exchange data at the same time. This situation is resolved by defining one of the sessions the contention-winner (often called the *conwinner*) session and the other the contention-loser (or *conloser*) session. The primary session partner refers to its session as the contention-winner session, and the secondary session partner refers to that

same session as the contention-loser session. The contention winner side of the session is the one that initiates the BIND.

CP-CP sessions are only established between adjacent APPN nodes and always use the CPSVCMG logon mode name and APPN class of service.

### 8.3.1 End node CP-CP sessions

An end node establishes CP-CP sessions with an adjacent network node. The network node that has a CP-CP session with the end node is referred as the network node server (NNS).

End nodes can have active links to many network node or end nodes at the same time, but can establish a CP-CP session pair with only one network node server at a time. If a CP-CP session fails, an end node can immediately choose another adjacent network node to act as its network node server. Note the following points:

- ▶ End nodes never establish CP-CP sessions with other end nodes.
- ▶ The end node is the node that always initiates the activation of the CP-CP session.
- ▶ End node can register their local topology (links to other end nodes and network nodes) with their network node server.

### 8.3.2 Network node CP-CP sessions

To perform directory services, topology and route selection services, adjacent APPN nodes throughout the APPN network use the set of two CP-CP sessions to exchange network and control information.

A network node or composite network node can establish CP-CP sessions with any network node or composite network node to which it has an APPN direct link that supports CP-CP sessions.

CP-CP sessions between two network nodes are used to perform searches for resources, exchange topology information, and can be used to register resources with a central directory server. After an APPN connection has been established, identification information is exchanged between the nodes, and CP-CP sessions can be established between the control points in the directly attached nodes. After the CP-CP sessions are established, the two nodes exchange CP capabilities, which indicate the level of network services provided by the control point.

### 8.3.3 NETID considerations

Each APPN node has one network ID and one CP name assigned. The network ID identifies the network to which the node belongs, and the CP name is unique within that network. The network ID and CP name are defined in the APPN node at the time of system definition. Within an APPN network, all interconnected network nodes share a common network ID.

CP-CP sessions are allowed only between network nodes that have the same NETID, unless one or both network nodes are defined as extended border nodes (see “Extended border node (EBN)” on page 175).

An end node can use the same network ID as its network node server, or it can use a different network ID.

## 8.4 APPN databases

APPN provides automatic network topology and directory support within APPN networks that simplifies network definition and permits dynamic selection of network routes. Some of the functions of APPN are topology and route selection services, and distributed directory searches, as described here:

### ***Topology and route selection services***

Topology and route selection services selects the best route to access a remote LU based on a set of user-specified criteria. Using the properties of the nodes and links in the network that are maintained in a network node topology database, a network node server calculates the best route from the local control point of the primary LU to the control point of the secondary LU according to the class of service selected by the LU initiating the session.

Topology and routing services is responsible for three functions:

1. Maintaining the topology database
2. Maintaining the LOGMODE-to-APPNCOS mapping table
3. Calculating session routes based on (1) and (2).

### ***Distributed directory searches***

Distributed directory searches determine (through a network search) the current node location of any remote logical unit (LU) that is known locally only by name. This alleviates the need to define routing or location information for every remote LU with which a local LU can establish a session. The information collected during the directory search is stored in the network node's directory database.

You can compare the directory services database to a telephone book or address book, in which you look up a name (an LU's name) and determine its address.

An SNA session is the logical connection between two LUs. The LU that originates the session is named the originating LU (OLU), and its session partner is the destination LU (DLU). As its name implies, the originating LU initiates the session by sending a BIND to the destination LU.

The topology database is like a map, in which the APPN network nodes furnish the two addresses (the locations of the originating logical unit, OLU, and destination logical unit, DLU) and decide what the best route is between them based on your "driving requirements" (or "class of service"). That is, do you want to take the high speed route, the scenic route, or the most direct route? The BIND sent by the originating logical unit uses the best route that is available at the time the session is setup.

### **8.4.1 Topology database (TOPO DB)**

The topology database consists of a local topology database, unique to a node, and a network topology database, whose entries are replicated across all network nodes in the same topology subnetwork. The topology database stores and maintains the nodes and the links (transmission groups or TGs) in the networks and their characteristics.

A component called the topology database manager (TDM) creates and maintains the topology database.

An APPN network node provides route selection services to itself and to its client end nodes. It maintains a network topology database that has complete and current information about the topology subnetwork or NETID subnetwork in which it resides. This information includes all the network nodes in the subnetwork and their node characteristics, and all the links interconnecting these network nodes and their link characteristics, thus creating a connectivity "map" showing the arrangement of nodes and links.

A network node uses the network topology database to compute routes for sessions that originate at LUs in its domain (that is, in itself and in client end nodes). Each route that a network node computes is the current least-weighted route from the node containing the origin LU (primary logical unit, PLU) to the node containing the destination LU (secondary logical unit, SLU) for the requested class of service. To determine an appropriate path through the network, the route selection algorithm first assigns weights to transmission groups and nodes. These weights are scalar values for each node and

transmission group, based on the relative significance of the characteristics for the requested class of service.

**Note:** The primary logical unit is the LU that sends the BIND.

An end node maintains only a local topology database, while a network node maintains both a local topology database and the network topology database. The network topology describes the network nodes and transmission groups between network nodes (the backbone). All network nodes have an identical copy of this data (other than transient differences while new topology information is in transit).

The local topology database stores the adjacent end nodes and the links to them. Changes in the local topology database are not propagated to other nodes.

End nodes have a limited topology database that is only used to establish CP-CP sessions to a network node server (NNS), and to inform the NNS of its links to other end nodes and network nodes.

## 8.4.2 Directory services database (DS DB)

An APPN network node provides directory services to its locally resident LUs and to the LUs in its client end nodes. It also assists other network nodes in the network in their search for resources. When network nodes learn the location of resources by searching, they cache the location of these resources in the local directory database.

The network node and the collection of resources it serves are called a *network node domain*. For an LU located in a network node, the local directory maps an LU name to the CP name of the network node where that LU is located. For an LU located in an end node, the directory includes the CP name of the owning end node and the end node's network node server.

Because APPN end nodes do not maintain a directory database, an end node that does not currently have CP-CP sessions with a network node server cannot establish sessions with resources located on other nodes unless these resources are predefined in a manner similar to low-entry networking partner LU definitions.

A low-entry networking node or APPN end node maintains a local directory containing entries for locally resident LUs. An APPN end node that does not currently have CP-CP sessions with a network node server also maintains entries for those resources residing in an adjacent node connected as a peer, such as another end node.

Entries in a low-entry networking node are defined manually. Because it does not support CP-CP sessions (it cannot have a network node server), a low-entry

networking node sends an LU-LU session activation (BIND) request over the link associated with the predefined session partner. If the destination LU is located in a node that is not adjacent to the low-entry networking node, it must be connected to a network node. This network node can locate the resource and select the appropriate route.

An APPN end node has an alternative to the low-entry networking's complete directory of all LUs with which it initiates sessions: it can initiate a locate search into the APPN network to find a desired LU by invoking the services of its network node server. Because the network node server identifies the route to be used for the requested session in its search reply, the resulting LU-LU session is not required to traverse the network node server.

## 8.5 APPN network topology

Each network node starts with a topology database containing only itself. When it joins the network, a network node receives a copy of the current network topology database through a topology exchange with another network node in the network. As long as it remains connected to the network, its local copy of the network topology database is updated whenever changes occur to intermediate routing resources within the network to maintain consistency with the topology databases of all other connected network nodes.

### 8.5.1 Topology data update (TDU) flows

Figure 8-8 on page 185 represents an APPN with three network nodes (NN2, NN3, and NN4). NN1 is about to join the APPN Network. Prior to connecting NN1 to the APPN network, every network node has an identical representation of the network in its topology database. NN1 starts with a topology database containing only itself.

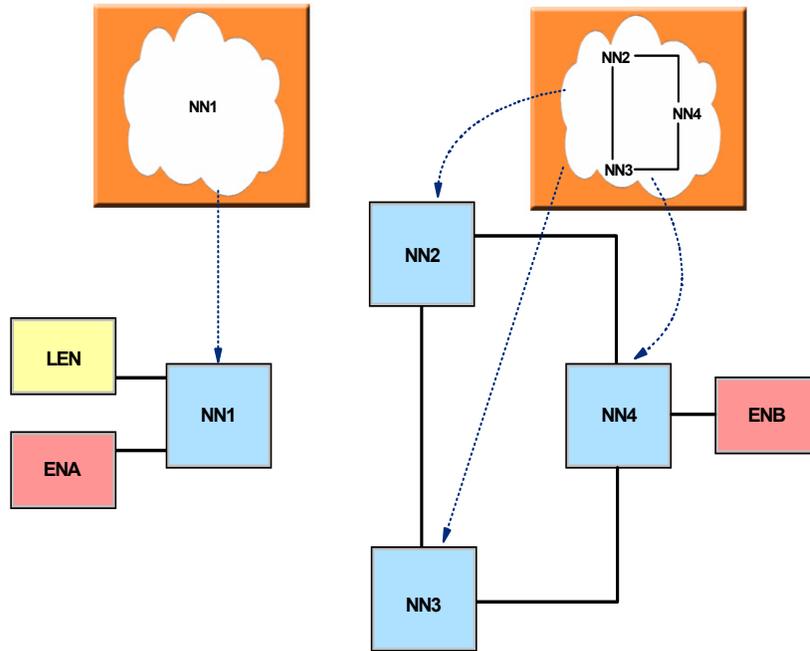


Figure 8-8 Network node joins the APPN network

When NN1 joins the network, topology data updates are sent over the CP-CP sessions. These topology database updates are sent whenever a node or transmission group state or characteristic changes or when CP-CP sessions are started or ended.

The topology database updates are propagated to all other adjacent network nodes; see Figure 8-9 on page 186.

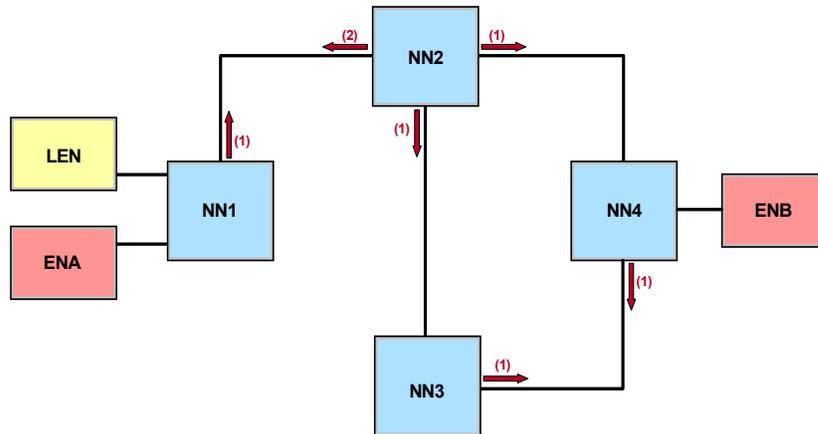


Figure 8-9 Topology database update (TDU) flow

TDU (1) is originated by NN1 and is sent from NN1 to NN2. NN2 propagates TDU (1) to all its adjacent network nodes (NN4 and NN3). The adjacent nodes that received the TDU (1) propagated TDU to their adjacent network nodes.

NN2 sends TDU(2) that describes the entire network topology, including the transmission group from NN2 to NN1.

As you can see, NN3 receives two identical TDUs (1), one from NN4 and one from NN2. How does NN3 determine whether to update its database when it receives an identical TDU? NN3 uses the *resource sequence number*, as explained here.

### **Resource sequence number (RSN)**

A resource sequence number is associated with each transmission group and node record in the topology database. Thus, when receiving resource updates in a topology database update, a network node can determine if the information in the update is new or old by comparing the resource sequence number in the update request with the resource sequence number of the stored record in the topology database (if the record has already been created).

If the resource sequence numbers are equal or the resource sequence number in the update is less than the resource sequence number in the stored record, the information is old and the network node does not need to update the topology database with the updated record and also does not need to forward this resource update in a topology database update to its network node partners. Only new information needs to be forwarded.

There is also the *flow reduction sequence number*, as explained here.

### **Flow reduction sequence number (FRSN)**

Network nodes assign flow reduction sequence number (FRSN) values to topology resource records when the records are modified or newly created. The assigning of a FRSN number to topology database updates and topology resource records allows a network node to track the most recent time the local network node sent a topology database update to each of its network node partners, as well as the most recent time each resource record in its topology database was last modified and broadcast in a topology database update.

The FRSN identifies how much of the topology database must be exchanged when the network node rejoins the network.

Once all topology database updates have been sent and received, all network nodes have the same view of the network. Figure 8-10 depicts the topology database that exists in all network nodes once topology database has reached “steady state.”

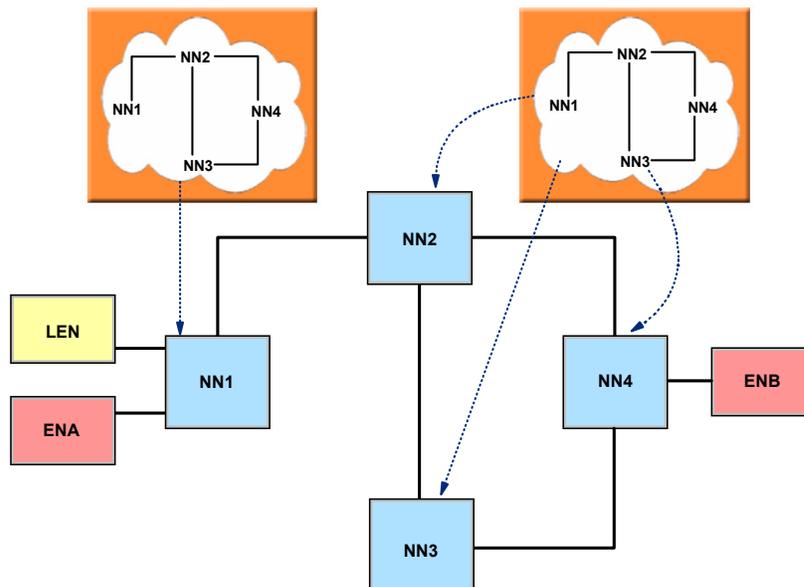


Figure 8-10 APPN network topology - steady state

## **8.5.2 Locating resources**

Among the functions that directory services in an APPN network node provide is finding the location of a specified destination resource. This is done using the following steps until one succeeds:

- ▶ Search the local directory database for the location of the destination resource. If the location of the destination resource is known (or thought to be known) a directed search is sent to the suspected destination.
- ▶ If no location information about the destination resource is found in the directory database (or if the directed search to the suspected destination fails because the resource has moved or not longer exists), perform a broadcast search of served end nodes.
- ▶ If the destination resource still has not been located, this network node performs either a central directory server (CDS) search or a broadcast search of the native network.
- ▶ If this network node is not a central directory server, then the topology database is used to determine if a central directory server exists in the network. If a central directory server exists, then a directed search is sent to the central directory server to determine the location of the resource. The central directory server is then completely responsible for locating the resource in the native network and any attached subarea networks. This is done by first performing a broadcast search of the native network and, if necessary, a serial (directed) search of all interchange nodes in the network to allow these ICNs to search their attached subarea network(s).
- ▶ If this network node is a central directory server or if no central directory server exists in the network or if the directed search to a central directory server fails to reach the central directory server, then this network node is responsible for locating the resource in the native network and any attached subarea networks. This is done by first performing a broadcast search of the native network and, if necessary, a serial (directed) search of all interchange nodes in the network to allow these ICNs to search their attached subarea network(s).
- ▶ If the destination resource still has not been located and this node is a border node, perform a serial search of adjacent APPN networks by sending directed searches to other border nodes.

### **Broadcast search**

A broadcast search is issued by a network node because a directed search using database information has failed or there is no database information for the requested resource. A broadcast search does not use database information about the location of a requested LU to propagate the search. Instead, a broadcast search is sent to every adjacent network node at the same time. Each of the adjacent network nodes then forwards the broadcast search to all other adjacent network nodes, and so forth. After propagating a broadcast search to all adjacent network nodes, each network node also searches all its client end nodes to determine if the target LU resides within the domain of the network node.

This process allows the entire network to be searched. When the search reaches the network node serving the destination resource, that node sends back a positive reply to the first search request it receives.

In Figure 8-11, LUA on ENA wants to start a session with LUB on ENB.

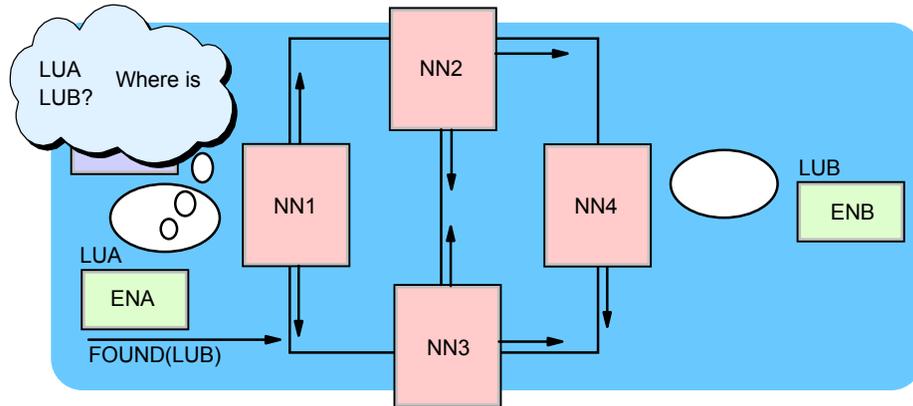


Figure 8-11 Broadcast search - stage 1

ENA sends a directed search to its network node server (NN1). Because the directory database of NN1 has no information about LUB, NN1 sends a network broadcast search to all end nodes served by NN1. If LUB is not located in the NN1 local domain, the broadcast search is propagated at the same time to all adjacent network nodes.

After propagating the broadcast search, network nodes search local node and served end nodes.

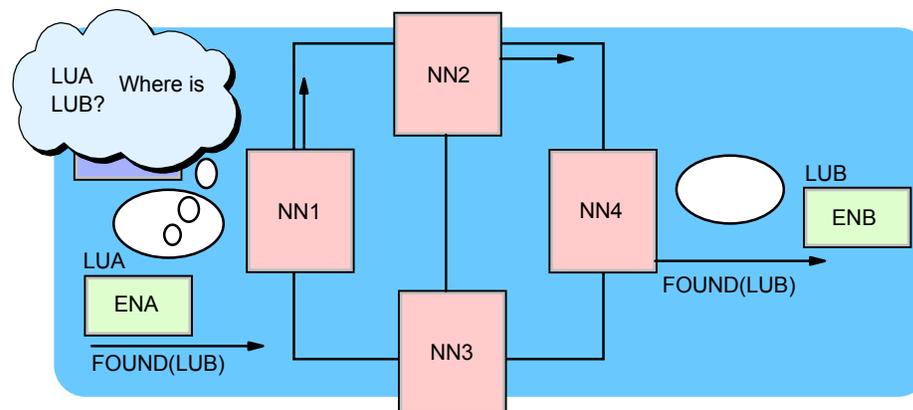


Figure 8-12 Broadcast search - propagating to served end nodes

Since LUB is located in ENB, ENB sends a positive reply to the broadcast search request. The reply is returned along the same path as the request. The directory database in NN4 and NN1 is updated with the information about the locations of both LUB and LUA.

### **Directed search**

A directed search is sent by directory services to the node recorded as the owner of the requested LU to verify the information. A directed search can be sent to a network node server from an end node and to an end node from a network node server. A directed search can also be sent from a network node to a central directory server when the network node does not have information on the location of the destination logical unit or following a failed search.

Directed searches are always sent to the network node server of the destination resource. This is because the topology database does not allow a network node to compute a Locate path all the way to an end node in the domain of another network node (that is, to an end node that is served by a different network node).

A directed Locate search request is a request that is sent along a predefined path from one network node to another network node. The origin network node calculates a path of CP-CP session hops to the target network node and appends the routing information to the search. Each network node along the path relies on that routing information for choosing the next hop and ensuring that the search travels directly to the destination network node.

Figure 8-13 on page 191 demonstrates how a network node (NN4, in the example) verifies the location of LUA. The request to locate LUA originates from ENB, which sends a locate (FIND) request to its network node server. The directory database in NN4 has the information about the location of LUA and sends a directed search. Because LUA is located in an end node (ENA), the directed search is addressed to ENA's network node server (NN1).

The search contains the path over which the directed search should be sent. Each network node along the path forwards the directed search to the next network node. The last network node on the path (NN1) forwards the search to the end node (ENA).

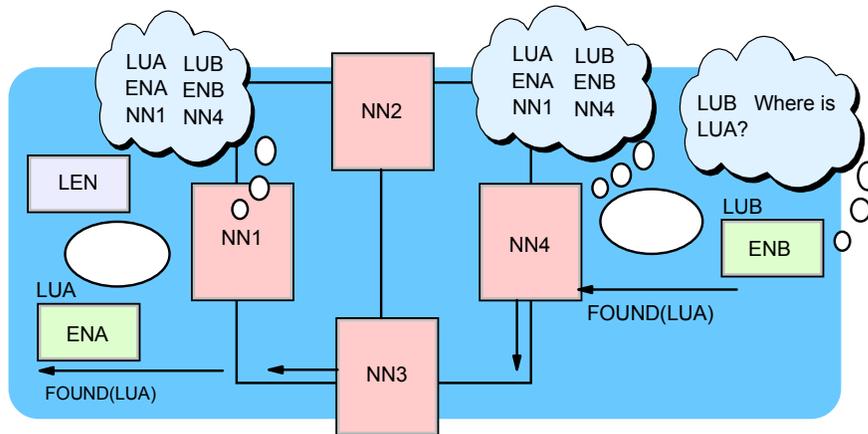


Figure 8-13 Directed search - verifying resource location

The positive reply is returned along the same path as depicted in Figure 8-14.

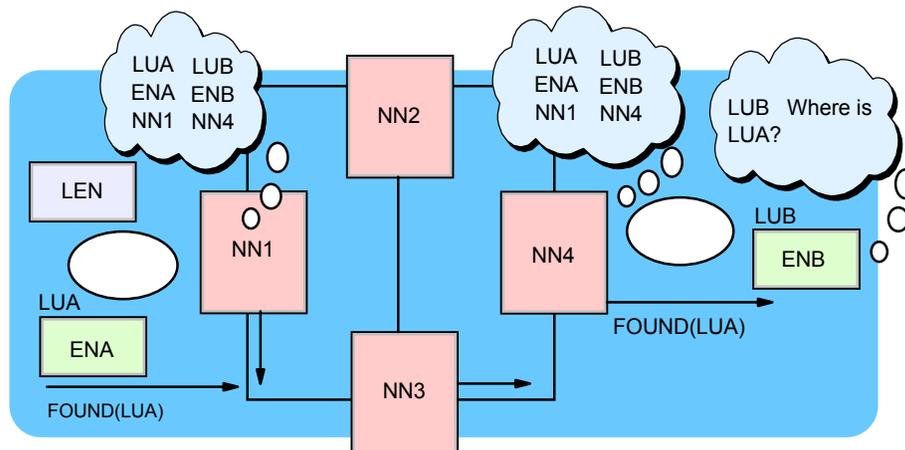


Figure 8-14 Directed search - positive reply

### Using a CDS to minimize broadcast searches

The function and services of the central directory server are discussed in “Central directory server (CDS)” on page 174. The following is an example of another search that illustrates the role of a central directory server.

In Figure 8-15 on page 192, NN1 is assigned as the central directory server for the APPN network. LU2, which resides in NN2, wants to start a session with LUB.

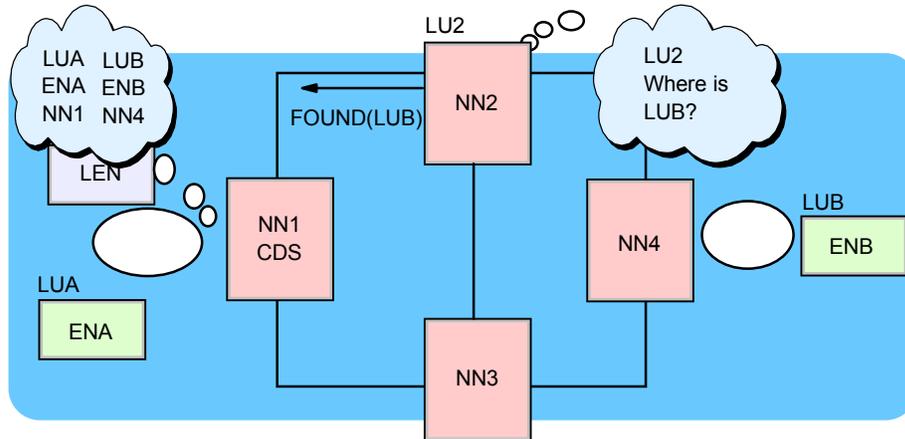


Figure 8-15 Central directory server example - LU2 to establish a session with LUB

Neither NN2 nor NN3 saved the location of LUA or LUB during the examples of broadcast search and directed search. NN2 and NN3 did not issue the search, they just forwarded the search. For the broadcast case, NN2 and NN3 must “interpret the search” in order to determine if the target LU is in their domain.

The reason these intermediate network nodes do *not* cache this information is because they may never need it! Because LUB is not stored in the directory database, the only way to locate LUB is to initiate another broadcast search.

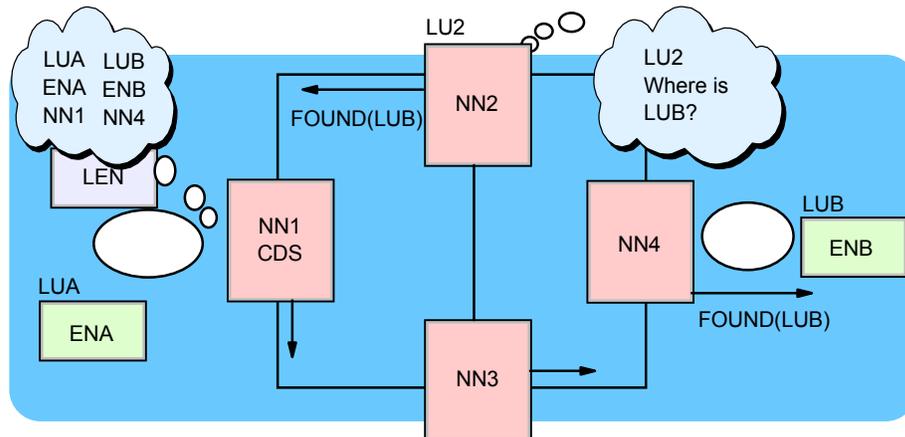


Figure 8-16 NN1 is the central directory server

Instead of initiating a broadcast search, NN2 sends a directed search to the central directory server. The central directory server is now completely responsible for locating LUB in the native network.

This may include:

- ▶ Sending a directed search to verify the location of LUB
- ▶ Sending a directed search queries to other central directory servers in the network
- ▶ Sending a broadcast search to determine the location of LUB

If a broadcast was necessary, then the central directory server will cache the location of both the originating LU and destination LU (if found). This is how a central directory server builds up knowledge of all of the resources that existing the network (not just resources that reside in their domain and their session partners).

As in the previous examples of directed search and broadcast search, the positive reply is returned to the central directory server along the same path that the directed search followed. NN2 now caches the location of LUB (because it was the network node server (originating LU)) for this search

Here, there is one more step. Since the positive reply was sent to the central directory server, the central directory server has to inform NN2, which initiated the search, that the resource exists and its location.

### 8.5.3 Route calculation

During session establishment, the network node server of the origin LU refers to the topology database to calculate and select the current best route through the APPN network from the primary LU to the secondary LU for the requested class of service. A network node calculates routes for sessions that originate at the LUs in it and at the end nodes it serves. When a route is calculated, it is stored and can be reused.

Route selection is based on how the actual characteristics of each node and transmission group along the possible paths match the characteristics required by the requested class of service. The route that a network node selects is the current least-weight (or best) route from the node containing the origin LU to the node containing the destination LU.

Because remote end node links are not stored in the topology database, route calculation for sessions between LUs that reside in end nodes uses transmission group vectors (TGV). The end node sends a transmission group vector for each link that it has to other network nodes or end nodes on a locate request or reply. These end node transmission group vectors are temporarily added to the topology database when computing routes to or from end nodes.

## 8.5.4 Transmission group (TG) characteristics

For each APPN transmission group, transmission group characteristics are defined. To distinguish one transmission group from another, transmission group characteristics can be specified in the following ways:

- ▶ As transmission group profiles, which are groups of transmission group characteristics that can be applied to several PUs
- ▶ On definition statements

The characteristics of transmission groups owned by other nodes are learned through topology database updates.

Eight standard transmission group characteristics are defined by the APPN architecture. These transmission groups must be used by all APPN products. The transmission groups are: COSTTIME, COSTBYTE, PDELAY, CAPACITY, SECURITY, and the three User Parm values (UPARM1, UPARM2 and UPARM3).

Each product implementing APPN can decide what transmission group characteristics should be for various APPN links. Most APPN products (including VTAM) try to assume reasonable default values based on the type of the link.

## 8.5.5 Class of service (CoS)

An APPN class of service (CoS) defines the required or requested characteristics of a route for a session. A *class of service* consists of a set of ranges of acceptable values for the characteristics of links and nodes to be used for a session specifying that particular class of service.

APPN classes of service are defined in a VTAMLST definition list. Unlike the class of service for the subarea network, where the class of service is actually a list of VRs that are acceptable for a particular class of service, APPN class of service specifies the types of routes that are acceptable for a class of service.

- ▶ Each APPN class of service has a table of definitions.
- ▶ Each column represents a transmission group or node characteristic.
- ▶ Each row represents a “class” of transmission groups or nodes; see Table 8-1 on page 195.

Table 8-1 #CONNECT class of service LINEROW values

LINEROW	CAPACITY	COSTBYTE	COSTIME	PDELAY	SECURITY	UPARM1UP ARM2UPAR M3	Weight
1	4M MAXIMUM	0 0	0 0	MINIMUM NEGLIGIB	UNSECURE MAXIMUM	0 255	30
2	56000 MAXIMUM	0 0	0 0	MINIMUM TERRESTR	UNSECURE MAXIMUM	0 255	60
3	19200 MAXIMUM	0 0	0 0	MINIMUM TERRESTR	UNSECURE MAXIMUM	0 255	90
4	9600 MAXIMUM	0 0	0 0	MINIMUM TERRESTR	UNSECURE MAXIMUM	0 255	120
5	19200 MAXIMUM	0 0	0 0	MINIMUM PACKET	UNSECURE MAXIMUM	0 255	150
6	9600 MAXIMUM	0 128	0 128	MINIMUM PACKET	UNSECURE MAXIMUM	0 255	180
7	4800 MAXIMUM	0 196	0 196	MINIMUM MAXIMUM	UNSECURE MAXIMUM	0 255	210
8	MINIMUM MAXIMUM	0 255	0 255	MINIMUM MAXIMUM	UNSECURE MAXIMUM	0 255	240

Note the following:

- ▶ Each row is defined by (min/max) range for each characteristic.
- ▶ Each row defines a weight for transmission groups or nodes that fit the range.
- ▶ Rows are typically defined from most to least restrictive (low weight to high weight).

## 8.5.6 Choosing session paths

VTAM chooses a route by comparing the actual characteristics of the available nodes and transmission groups to the allowed characteristic ranges specified in the requested class of service. For each APPN class of service entry, there are 1 to 8 LINEROW operands and 1 to 8 NODEROW operands. These operands give up to 8 acceptable sets of characteristics for the lines and up to 8 for the nodes in each class of service.

The WEIGHT parameter on the NODEROW and LINEROW operands is coded to indicate the desirability of that set of characteristics. The lower the value of the

WEIGHT parameter, the higher the desirability of a node or transmission group that fits that set of characteristics.

The values shown in Table 8-1 for capacity, cost per byte, cost per unit of time, propagation delay, security level, and the user-defined characteristics (UPARM1, UPARM2, and UPARM3) represent ranges, with the top value in a LINEROW representing the minimum value and the bottom value in a LINEROW representing the maximum value. These values are compared the total path weight (links and nodes).

## 8.6 High performance routing (HPR)

High performance routing (HPR) is an addition to APPN that improves reliability, increases network performance, and was designed to exploit higher link speed technologies.

Intermediate session routing (ISR) requires significant processing for error control, flow control, and segmentation at each intermediate node. The significant processing causes significant latency in each node.

As higher speed connections evolved, the APPN architecture was required to introduce some changes and enhancements to allow switching in intermediate nodes to be done at higher speeds (that is, lower layers) thereby improving the throughput of data.

HPR addresses this by routing at layer 2 and 3 and changing the existing intermediate session routing (ISR) which is done in basic APPN at layer 5. HPR introduced new headers (see 8.6.3, "HPR headers" on page 200) that HPR analyzes to determine the next hop to route the message. Inspecting headers of the higher layer requires more resources, and that affects the performance. As HPR is done in lower layers than ISR, the delay in each node along the path is shortened.

HPR has also shifted the error recovery to the end points, instead of individual lines. The two endpoints are the APPN nodes, end node or network node, that provide for the LU- LU session. With basic APPN, every network node was responsible for recovering from errors on the two links that were used to deliver the data to and from the network node. The error recovery consumed resources and affected performance.

With high speed networking, the reliability of the communication lines improved dramatically. Today the ratio of errors-to-traffic is in the range of 10<sup>-9</sup>. The probability for error is very low and moving the responsibility for error recovery to

the end points improves performance and does not affect the integrity of the data.

HPR has also been designed to provide a non-disruptive path switch to route around failures. In simple words, non-disruptive path switching addressed one of the major deficiencies of SNA compared to other protocols. With non-disruptive path switching, a session is switched to another available path without affecting session availability to the end user.

In the following section we discuss the two major components of HPR, the rapid transport protocol (RTP) and automatic network routing (ANR).

### 8.6.1 Rapid transport protocol (RTP)

RTP is a connection-oriented, full-duplex protocol designed to support data in high-speed networks. RTP connections are established within an HPR subnet and are used to carry session traffic. These connections can be thought of as transport pipes over which sessions are carried.

RTP connections can carry data at very high speeds by using low-level intermediate routing and minimizing the number of flows over the links for error recovery and flow control.

The RTP functions include:

- ▶ Non-disruptive path switching

An RTP connection's physical path can be switched automatically to reroute sessions around a failure in the network. The RTP connection is reestablished over a new physical path that bypasses the failing link or node, and the session's traffic flow is resumed on the RTP connection non-disruptively. Any data that was in the network at the time of the failure is recovered automatically using RTP's end-to-end error recovery.

- ▶ End-to-end error recovery

In basic APPN, error recovery is done on every link in a network. To address the emerging high-speed lines with much lower bit error rates, HPR removed the requirement to do link-level error recovery and instead does error recovery on an end-to-end basis. This improves performance by reducing the number of flows required to do the link-level error recovery on every link. RTP also supports selective retransmission, where only missing or corrupted packets are re-sent, and not all packets since the failure occurred.

- ▶ Resequencing packets

A major observable fact in a multilink transmission group (MLTG) is that packets may arrive at the endpoint out of sequence. The RTP endpoints resequence the data in this case.

► End-to-end flow control and congestion control

Flow control is the mechanism that controls the pace at which data is sent into the network to prevent flooding the resources along the route and to prevent the endpoint from being congested. In an APPN network, flow control is done on each stage of the session by using adaptive session-level pacing. This method provided excellent performance for networks with low speed lines and poor quality. For high-speed networks, adaptive session-level pacing was found inadequate due to the amount of processing required in each node.

HPR introduced a protocol that is suited for high-speed routing called adaptive rate based (ARB) flow/congestion control. It regulates the flow of traffic over an RTP connection by adaptively changing the sender's rate based on feedback from the receiver. This protocol allows for high link utilization and prevents congestion before it occurs, rather than recovering from congestion after it occurs.

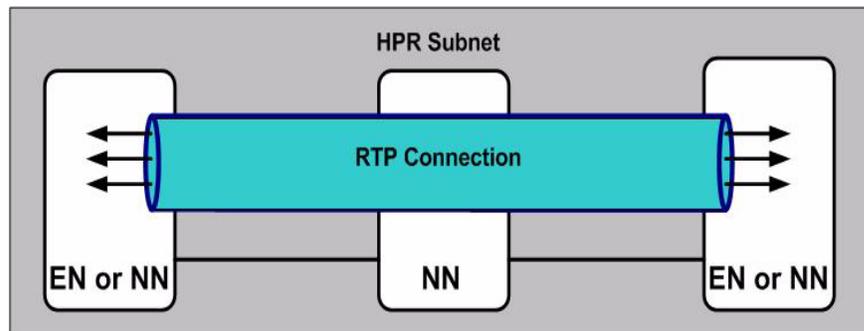


Figure 8-17 RTP connection

Figure 8-17 shows an RTP connection that is carrying multiple sessions. Traffic from many sessions requesting the same class of service can be routed over the same RTP connection.

## 8.6.2 Automatic network routing (ANR)

Automatic network routing (ANR) is a low-level routing mechanism that minimizes cycles and storage requirements for routing packets through intermediate nodes. An ANR node is an intermediate network node on the path of an RTP connection. ANR nodes are not aware of SNA sessions or RTP connections passing through the node. All an ANR node must do is read the header in a network layer packet and forward the information to the next node on the path. The ANR information is learned by the RTP endpoints during establishment of the RTP connection by sending a "Route Setup" message which flows through all nodes on the prospective HPR path.

The ANR functions and services include:

- ▶ Fast packet switching

ANR takes place at a lower layer than APPN intermediate session routing and significantly improves performance in the intermediate nodes. The ANR node routes the HPR packet and does not provide functions such as link-level error recovery, segmentation, flow control, and congestion control. These functions are performed at the RTP connection endpoints.

- ▶ No session awareness

Intermediate nodes that implement ANR are not aware of the SNA sessions or the RTP connections that are established across the nodes. This means that there is no requirement to keep the routing tables for session connectors that are kept in basic APPN.

- ▶ Source routing

Source routing is a technique whereby the sender of the data specifies the route that the data should take through the network. In an IP network (which is a connectionless network), every router examines the data's header and selects the next hop. In an ANR network, the end node or the network node selects the route and the network nodes along the route forward the data based on selection of the originator.

ANR is a source-routing protocol and carries the routing information for each packet in a network header with the packet. Each node strips off the information it has used in the packet header before forwarding onto the link, so the next node can easily find its routing information at a fixed place in the header. This means that switching packets through a node can be done more quickly than in the routing table lookup method used in basic APPN. There is no restriction on the number of hops in ANR.

Figure 8-18 shows how ANR routes messages.

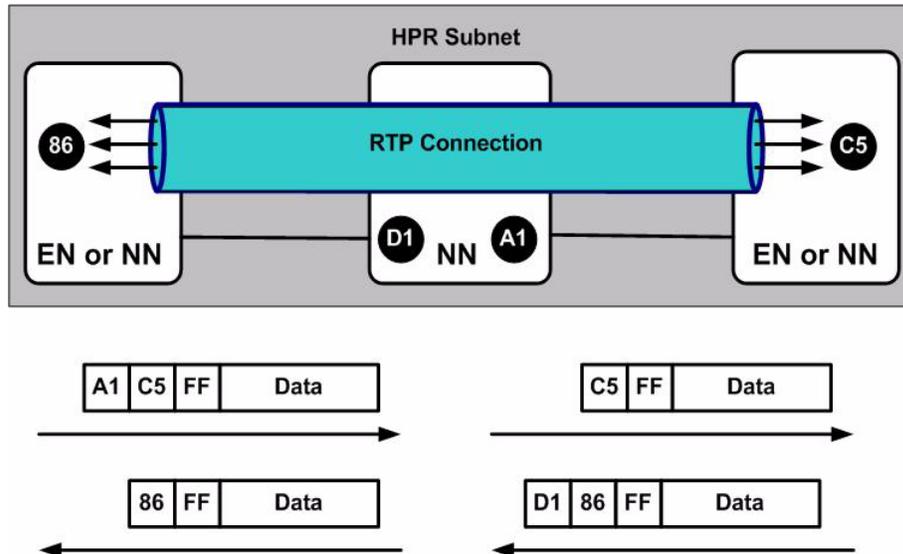


Figure 8-18 ANR routing

In the figure, the intermediate network node receives the message and strips the first routing label (A1) from the message header before forwarding the packet on link A1. The address of C5 represents the endpoint in the last HPR node. Since the routing is pre-determined, the intermediate network node routes the packets very quickly, with no need to examine the request unit (RU) and/or the transmission header (TH), reserve storage or buffers, or to do link-level error recovery. The same mechanism takes place on the reverse route.

### 8.6.3 HPR headers

HPR adds to the basic header in the SNA path information unit (see "Path information unit (PIU)" on page 149). The packet transported along an RTP connection has a specific format. It consists of three components: the network layer header (NHDR), RTP transport header (THDR, and data; see Figure 8-19 on page 201.

The network layer header begins the frame used by rapid transport protocol nodes. It provides addressing for the packet as it transverses the HPR network. The components of this header include the transmission priority and the ANR (automatic network routing) labels. The NHDR consists of some indicators that identify the packet as a network layer packet.



Figure 8-19 HPR header

## 8.6.4 Path switching

RTP can switch automatically and reroute data around a failed node or link without disrupting the LU-LU sessions. This is called *nondisruptive path switching* because LU-LU sessions survive link failures. Nondisruptive path switching within the HPR portion of the network automatically occurs in an HPR subnet to bypass link and node failures if an acceptable alternate path is available.

Link failure is detected by the RTP end points when the first link on the RTP pipe fails or data packets that were sent are not acknowledged. One or both RTP endpoints detect the failure and redrive the RTP setup. If necessary, another APPN search is initiated to locate the best path available to the partner LU.

Figure 8-20 shows a failed RTP and the alternate RTP that traverses different ANR nodes.

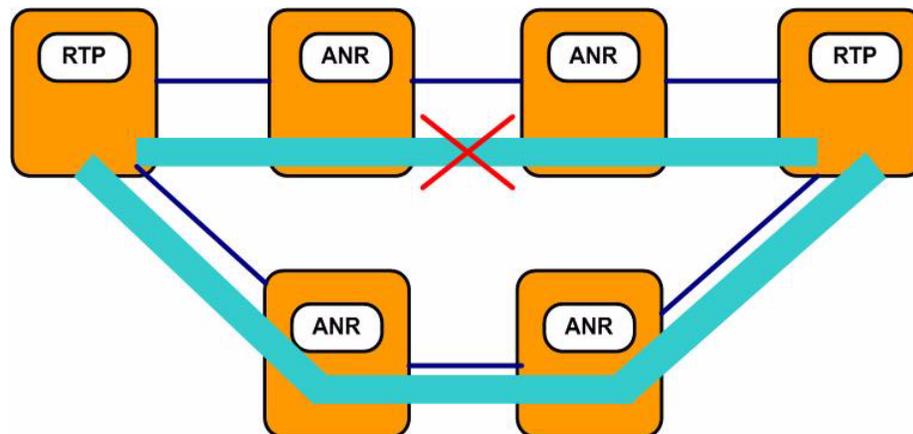


Figure 8-20 Non-disruptive path switching

## 8.7 Connection networks

A connection network is a representation of a shared access transport facility (SATF), enabling dynamic direct connectivity between any pair of link stations attaching to the facility. An example of a shared access transport facility is a local area network (LAN) where two nodes can communicate directly with each other without the need to use a router.

In Figure 8-21 there are end nodes and one network node connected to a LAN. If you do not implement a connection network, you have two options. The first option is to define links only from each end node to the network node (the left side of the figure); the second option is to implement a meshed topology by defining links between each pair of nodes (the right side of the figure).

The first option requires fewer link definitions (each end node must define only one link to the network node, and the network node must define a link to each end node, resulting in  $2*(n-1)$  link definitions for a network consisting of  $n$  nodes); but sessions between any two end nodes must traverse the network node, which can cause performance problems in the network node if there are a large number of nodes on the LAN.

The second option requires more link definitions (each node must define links to every other node, resulting in  $n*(n-1)$  definitions for a network of  $n$  nodes); but sessions between any two nodes always use a direct link between the session endpoints.

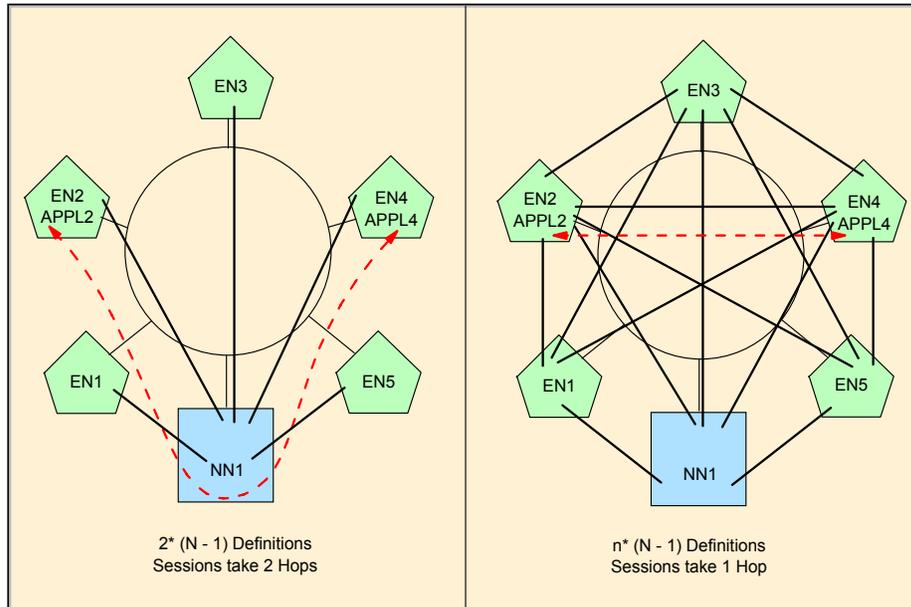


Figure 8-21 APPN transmission group definitions without a connection network

Also notice that, as more nodes are added to the shared access transport facility, the number of link definitions required grows *exponentially* with option 2, while the number of link definitions grows *linearly* with option 1. Another problem with option 2 is that, as more nodes are added to the LAN, new links must be defined on every other node on the LAN in order to exploit direct connectivity to the nodes being added.

The administrative task of defining new links is considerably reduced (and the overhead of routing session traffic through an intermediate network node is completely eliminated) by using a connection network to represent the shared access transport facility. Using a connection network allows nodes attached to the shared access transport facility to exploit direct connectivity over the LAN without defining a direct link to every other node on the LAN.

A connection network is implemented by defining a virtual routing node (VRN) to represent the shared access transport facility; see Figure 8-22. Each end node on the LAN defines a link to the real network node (for sending searches) and a link to the VRN which identifies the end node's connectivity to the shared access transport facility.

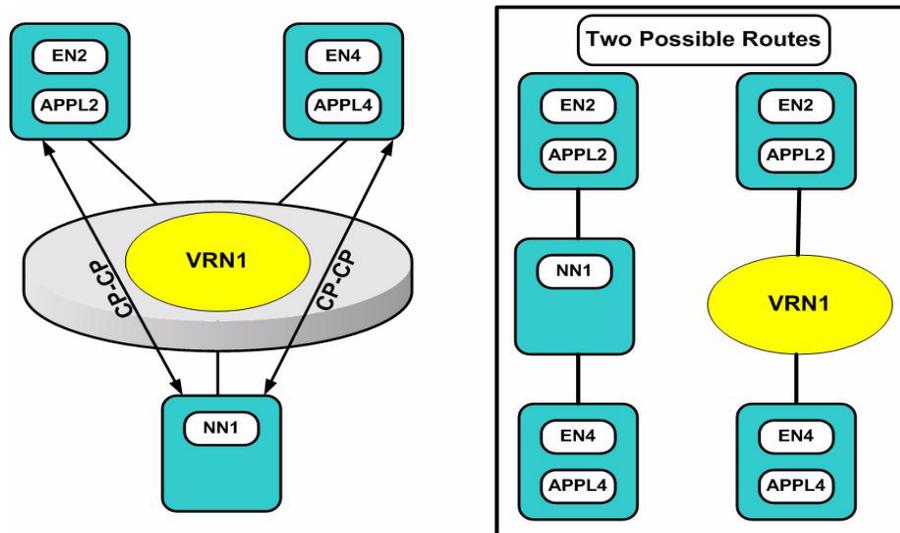


Figure 8-22 Example of a virtual routing node (VRN)

The links to the VRN provide the appearance of a second path between any two end nodes on the same shared access transport facility—but this second path is preferred over the path through the network node because the VRN is a virtual network node rather than a real intermediate network node. The VRN link definitions are used during session establishment to dynamically activate a new link directly between the session endpoints. When the last session traversing a dynamic VRN link ends, the dynamic link is inactivated until it is needed again for another session.

Because each end node attached to the connection network defines only 2 links, the number of link definitions required for a network with  $n$  nodes is  $2 \cdot n$ . This is very comparable to option 1 above, but avoids the overhead of routing sessions through real intermediate network nodes. Furthermore, when new end nodes are added to the connection network, new link definitions may be required on the network node; but no additional link definitions are required on any of the existing end nodes in order to exploit direct connectivity to the new end nodes.

## 8.8 Dependent LU requester/server (DLUR/DLUS)

Traditional SNA networks consist of peripheral nodes, called *physical units* (PUs), containing several types of logical units (LUs). The physical units are typically type PU2.0 (for example, a 3174 control unit) or type PU2.1 (such as a Windows-based workstation, AIX, or iSeries, which runs applications, gateways, or servers). The LUs (in PU2.0 or PU2.1 nodes) can be type LU0, LU1, LU2, or

LU3. Types LU0, LU1, LU2 and LU3 are called “dependent LUs” because they require VTAM services to establish LU-LU sessions.

Another LU type is LU6.2. An LU6.2 can act as either a dependent or independent LU. LU6.2 (APPC) acts as a dependent LU when it resides in a type PU2.0 node and as an independent LU in type PU2.1 nodes. As an independent LU, an LU6.2 can initiate sessions on its own (is capable of sending BINDs) without services from VTAM.

Historically, type PU2.0 and PU2.1 nodes have been attached to a subarea boundary function, typically to an NCP. As an alternative, many customers have used Open Systems Adapters, channel-attached routers, or 3172s to attach these devices to VTAM’s boundary function, rather than NCPs.

The dependent LU server (DLUS) is implemented only in Type 5 (VTAM) network nodes. The DLUS function enables VTAM to provide SSCP services for dependent LUs located in remote APPN end nodes or network nodes, which act as the dependent LU requester (DLUR).

Two LU 6.2 sessions (one inbound, one outbound) are established between a DLUS and a dependent LU requester (DLUR), and these LU 6.2 sessions are collectively known as the *CPSVRMGR pipe*. SSCP-PU and SSCP-LU session flows use the CPSVRMGR pipe. An SSCP-PU session is established between a VTAM network node and the PU that owns the dependent LU, and an SSCP-LU session is established between VTAM and the dependent LU.

Session initiation flows for the dependent LU are sent over the SSCP-LU session, and VTAM can use subarea or APPN flows to initiate a session with the PLU. BIND and session data are then routed directly between the PLU and the dependent LU.

Figure 8-23 illustrates a skeleton configuration of DLUR/DLUS.

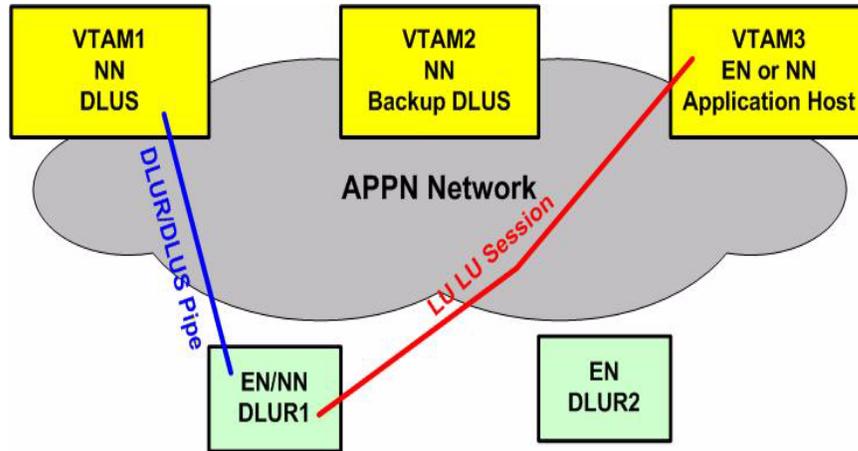


Figure 8-23 DLUR configuration

Note the following points:

- ▶ The DLUR can be located in remote sites. Using a DLUR also eliminates the need for dependent LUs to be physically adjacent to a VTAM or NCP subarea node.
- ▶ Dependent LUs attached to a DLUR can also exploit HPR, allowing their sessions to be non-disruptively routed around failures in the network.
- ▶ Session paths do not need to include an owning VTAM.

## 8.9 Defining a VTAM APPN network

The technique to define a VTAM APPN network is like the one used to define the VTAM subarea network in Chapter 7, “Systems Network Architecture - basics and implementation” on page 133. The definitions are put together in the ATCSTRxx in VTAMLST and by defining various VTAM major nodes.

### VTAM startup JCL

The first step for implementing APPN on a mainframe is to define the datasets used for APPN checkpointing and to update the VTAM startup JCL.

```
i/** DATA SETS FOR APPN DATABASE CHECKPOINTING
//DSDB1 DD DSN=SYS1.DSDB1,DISP=SHR
//DSDB2 DD DSN=SYS1.DSDB2,DISP=SHR
//DSDBCTRL DD DSN=SYS1.DSDBCTRL,DISP=SHR
//TRSDDB DD DSN=SYS1.TRSDDB,DISP=SHR
```

## Definitions in ATCSTRxx

1. Define the APPN node type. You need to assign the role of VTAM as an APPN node; that is, either a network node (NN) or an end node (EN). Do this by coding the NODETYPE parameter. The two possible values are either **NN** or **EN**.

To define VTAM as a network node, code the following:

```
NODETYPE=NN
```

2. Assign the control point name (CPNAME). The same parameter used for subarea networking (SSCPNAME) is used for APPN.

To assign CPNAME1 as the control point in ATCSTRxx, code:

```
SSCPNAME=CPNAME1
```

3. Assign a network ID (NETID). The same parameter used for subarea networking (NETID) is used for APPN.

To assign NET1 as the network ID in ATCSTRXX, code:

```
NETID=NET1
```

4. Specify the level of high performance routing support. VTAM can support either automatic network routing (ANR) or rapid transport protocol (RTP). The HPR parameter specifies which functions of HPR, either ANR or RTP, or both, are implemented by VTAM.

To indicate that VTAM provide RTP-level HPR support, specify:

```
HPR=RTP
```

HPR=RTP is the default.

5. Specify the default link type. For a VTAM type 2.1 node the CONNTYPE parameter determines whether connections to adjacent type 2.1 nodes are established as a low-entry networking (LEN) node (with no APPN support) or attempted as an APPN connection.

To specify that the connection be attempted as an APPN connection, code:

```
CONNTYPE=APPN
```

6. Specify CP-CP support. The CPCP parameter specifies whether VTAM supports CP-CP sessions with adjacent type 2.1 node.

To specify that CP-CP sessions be supported on all connections to adjacent type 2.1 nodes code:

```
CPCP=YES
```

The CONNTYPE and CPCP parameters can be overridden on the physical unit definition statement as illustrated in the following example:

```
PU1    PU    ADDR=01,  
        ISTATUS=ACTIVE,
```

```
CONNTYPE=APPN,  
CPCP=YES
```

7. Define VTAM as the central directory server (CDS). If you want to define VTAM as a central directory server, use the CDSERVER start option. You do not need to define anything on the other network nodes. The other network nodes find out about the existence of the central directory server through normal topology exchanges.

To define VTAM network node as a central directory server, code:

```
CDSERVER=YES
```

8. Control security. To reduce the burden of static definitions, VTAM permits dynamic definitions of physical units, logical units, and adjacent APPN nodes. In the mainframe environment, many installations are reluctant to allow dynamic definitions of VTAM resources. The DYNADJCP parameter controls whether adjacent control points are allowed to be created dynamically.

To disable the dynamic definition of adjacent control points, code:

```
DYNADJCP=NO
```

9. The BN=YES start option is used to define VTAM as an extended border node (EBN). VTAM EBNs must also be NNs.
10. IVTAM is defined as an ICN or MDH, then SORDER is used to control the order these networks are searched.

## Defining VTAM major nodes

1. Define the adjacent control point major node. If DYNADJCP in ATCSTRxx is defined as NO, the ADJCP major node defines the adjacent control point that can establish a CP-CP session with this VTAM.

```
ADJCP1 VBUILD TYPE=ADJCP  
CP1 ADJCP  
EBN1 ADJCP NETID=NET2
```

2. Create a network node server list for end node. To create a network server for an end node, code a NETSRVR major node. The major node list the network nodes that are part of the network node server list.

```
NETSRVR1 VBUILD TYPE=NETSRVR  
CP1A NETSRVR NETID=NETA  
CP2A NETSRVR NETID=NETA  
NETSRVR
```

In the last entry, the nameless NETSRVR entry allows the end node to select any other known adjacent network node that meets the defined criteria as its network node server.

## 8.10 Summary

APPN is the IBM strategic SNA protocol in the mainframe. It is required for sysplex, Enterprise Extender implementation, and many other technologies.

APPN is dynamic in nature and reduces the amount of predefinition required in an SNA subarea network.

APPN/HPR was introduced in the mid-1990s and supports nondisruptive route switching for failure recovery and connectionless intermediate routing of packets.

APPN/HPR still maintains the class of service (CoS) and data flow control using a more advance pacing mechanism, adaptive session pacing.

In contrast to subarea networking, where special hardware and software (VTAM in the mainframe and the network control program in the 3745) are required for intermediate session routing, every node that can act as network node can perform routing of SNA packets.

Key terms in this chapter		
APPC	APPN	automatic network routing (ANR)
broadcast search	central directory server	connection network
control point (CP)	CP-CP session	dependent LU requester/server (DLUR/DLUS)
directed search	directory services	end node
extended border node	high performance routing (HPR)	interchange node
low-entry networking node	migration data host	network node
network node server	rapid transport protocol (RTP)	route selection
Topology and route selection services	CNN	shared access transport facility (SATF)
VRN	Branch Extender	





## SNA/IP implementation

**Objectives:** After completing this chapter, you will be able to:

- ▶ Explain the background and motivation behind integrating SNA and IP networks.
- ▶ Describe technologies and solutions applied within mainframe-based organizations for integrating SNA and IP networks.
- ▶ List the features and benefits of each of these technologies.
- ▶ Describe how a z/OS network administrator implements these technologies.

## 9.1 Background

In the early 1990s, enterprises began to implement router technology in their backbone networks. At that time, enterprise networks resembled the Tower of Babel; that is, many networking protocols were proprietary and were not able to communicate with each other. Among the dominant proprietary protocols were IBM's SNA, Digital Equipment Corporation's DECNET, Novel's IPX™, and Microsoft's NetBIOS.

Router vendors were enthusiastic about introducing their products, and competition concentrated on whose product supported the greatest number of protocols. This introduction of multi-protocol routers into an enterprise backbone network helped consolidate many networking protocols into one infrastructure, thus reducing the expenses related to communication lines.

But not long after implementing routers in their backbone network, network managers realized that it is not easy to control, maintain, and perform problem determination in a network that implements multiple protocols. Implementing only one protocol in the backbone network reduces the complexity of the network and the router. So, there needed to be a single protocol or set of protocols.

The decision on which protocol set to use was easy. At that time, the Internet usage was growing dramatically among home users and enterprises. Because of the Internet, IP, TCP, and UDP were the protocols of choice. Many computer and operating system vendors added IP and the application protocols to their products and allowed access to their proprietary hardware and software using TCP/IP.

Some networking protocols, including IBM subarea network, were non-routable. By *non-routable* is meant that, based on the protocol headers, a router cannot decide where to route the packet. Although APPN intermediate session routing and high performance routing are routable, the resources (CPU and memory) required for implementing these protocols is very high and almost impractical. To accommodate SNA in a router-based network, IBM designed and defined several protocols that allow using SNA subarea and APPN protocols in a router-based IP backbone.

### 9.1.1 Why preserve SNA applications?

Ideally, all applications use the same protocol. Because TCP is the de facto application protocol, TCP is always the best solution. Using TCP as the application protocol requires no protocol conversion or encapsulation, and runs on the IP network. This, however is rarely possible within a large organization that has a huge investment in SNA applications.

A transaction-oriented program is dependent on the underlying protocol it uses. The application programming interface is different if one uses SNA or TCP. Changing a transaction-oriented program from SNA to TCP requires a redesign of the communication part in the program and replacing the code that handles error recovery, exception processing, and many other tasks. Many computer shops are reluctant to enter into huge conversion projects, such as SNA to TCP, that fail to introduce new architectures, man-machine interfaces, and the like. Shops use their budgets to modernize the applications rather than convert SNA applications just for the sake of having a common infrastructure.

Conversion of existing SNA applications to IP-enabled applications proves to be uneconomical and very technical due to the complexity of the applications (remember, these have been developed over many years), lack of skills in converting SNA to TCP, and the time required to do such a conversion. A conversion project can have a high degree of cost and risk.

This chapter discusses how to preserve SNA applications and investment in endpoint hardware (PU 2.0 or 2.1, such as banking or retail branch servers), while converging onto an IP-based backbone.

## 9.1.2 SNA applications and integration methods

SNA applications can be divided into two categories: 3270-based applications, and application-to-application, as explained here.

### 1. 3270-based applications

In this case, end users communicate with the mainframe using a 3270 display or a workstation that has 3270 emulation software installed. Data sent from a data communication (DC) program (like CICS, IMS, and sometimes even TSO) is displayed on the 3270 screen. The screen format is sent from the data communication transaction program and is not manipulated on its way to the 3270 screen.

### 2. Application-to-application

In this case, the remote end (branch office) has a programmable endpoint controller or server that does some local processing, sends the data using SNA to the data communication transaction program, and upon receiving the reply to an inquiry or database update, displays it on the end user's workstation.

In the early days of SNA, application-to-application communication used the LU 0 protocol, and as SNA evolved, new applications used APPC (see 8.1.1, "Advanced Program-to Program Communications (APPC)" on page 167).

There are several different ways of running mixed SNA and IP protocols over single IP protocol transport networks. The technique used to integrate SNA into the IP backbone depends on the SNA application type used.

**Telnet/3270** (Abbreviated as TN3270). This integration method is used for 3270-based applications. The SNA 3270 data stream is carried over TCP connections to a TN3270 server (z/OS). TN3270 is described in Chapter 10, “TN3270 Enhanced” on page 241 and is not discussed here.

**Data Link Switching** (Abbreviated as DLSw). SNA traffic is encapsulated in TCP packets.

**Enterprise Extender** (Abbreviated as EE). SNA/APPN (HPR) packets are encapsulated as User Datagram Protocol (UDP) packets over an IP network. HPR is discussed in Chapter 8, “SNA Advanced Peer-to-Peer Networking (APPN)” on page 165.

No change in the SNA application programs is required for any of the three integration methods.

Although DLSw can be used for other non-routable protocols, this book focuses on DLSw for SNA.

## 9.2 DLSw and Logical Link Control 2 (LLC2)

DLSw carries LAN traffic over the WAN. SNA uses the Logical Link Control 2 (LLC2). protocol for data transfer over the LAN. LLC2 is a connection-orientated layer 2 data link control protocol for SNA over a LAN.

### 9.2.1 Establishing an LLC2 connection over a LAN

When a LAN-attached station must establish an SNA session with a peer LAN-connected station, it sends an LLC2 broadcast message called TEST. TEST carries the MAC address of the destination station and originating station (see 2.3.2, “Media access control (MAC) addressing” on page 27). More data is carried in the TEST message, but this topic concentrates on the MAC address.

Since the TEST message is broadcast to all LAN-attached stations, every station examines the MAC address field in the message. If there is a match for the destination MAC address, the destination station replies with +RSP (positive response).

Figure 9-1 illustrates a connection attempt from workstation A (MAC address 4200.0000.0001) to workstation B (MAC address 4200.0000.0002). Note that the

TEST is a broadcast message and it is sent to *all* stations on the LAN—but for clarity, the other stations on the LAN are not shown here.

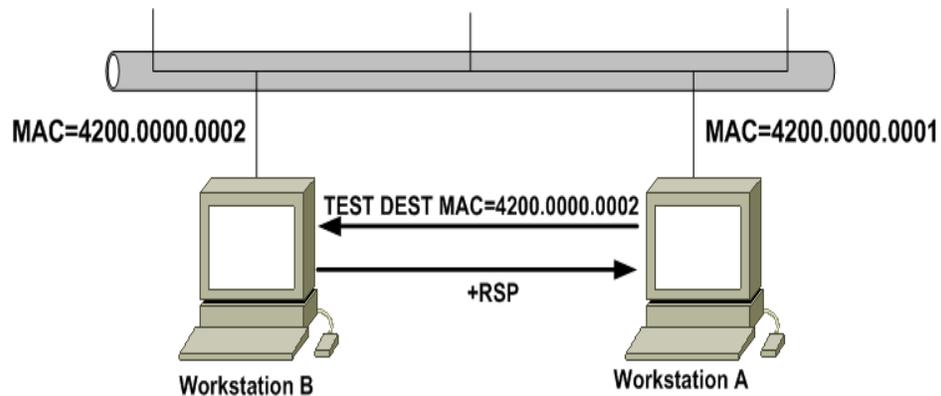


Figure 9-1 LLC2 search using TEST frame

After establishing the LLC2 connection between the two stations, the data transfer uses the LLC2 protocol.

## 9.2.2 Data link switching (DLSw)

Data link switching carries LAN traffic over the wide area network by encapsulating the LAN traffic in TCP packets. Data link switching was first developed by IBM to provide SNA support within multi-protocol routers. The DLSw formats and protocol were then made available to the wider community and published as RFC 1434. The RFC was then later enhanced and republished as RFC 1795.

Cisco's implementation of data link switching is known as DLSw+, and it contains additional enhancements to the original DLSw. The working infrastructures of most organizations include Cisco routers.

## 9.2.3 DLSw over the WAN

To illustrate how DLSw carries LAN (LLC2) traffic over the WAN, the distance between workstation A and workstation B is extended and two locations connect using a communication line and routers that implement DLSw. Figure 9-2 on page 216 depicts the new configuration.

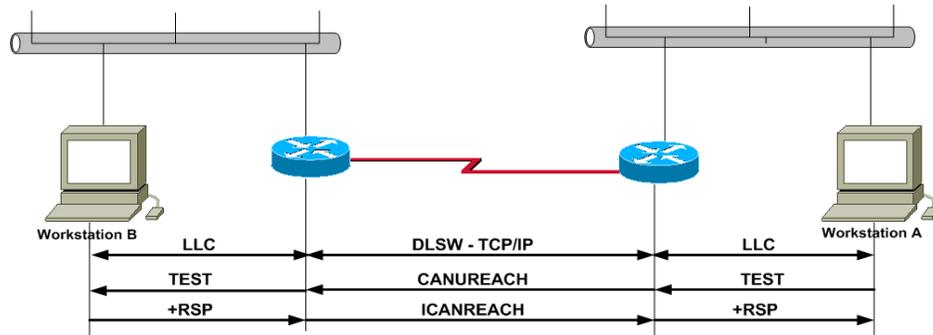


Figure 9-2 Connecting two LAN-attached workstations over a WAN using DLSw

As in the previous example, workstation A initiates a connection to workstation B by sending a TEST frame broadcast. Router A, which is attached to the same LAN, receives the TEST frame, encapsulates it in a TCP packet, and transmits it to all its peer DLSw routers. The TCP broadcast packet is called CANUREACH.

CANUREACH is received by all DLSw routers, including Router B. Router B has already learned the MAC addresses of the workstations, hosts, and servers that are attached to its LAN. When the CANUREACH packet with the MAC address of workstation B reaches router B, the router converts the CANUREACH packet to a TEST LLC2 frame.

Because the TEST is a broadcast frame, all stations on the LAN receive the TEST frame. Workstation B, which has been assigned the 4200.0000.0002 MAC address, responds with a +RSP. Because ICANREACH is a unicast message, Router B encapsulates the +RSP in a TCP packet and sends an ICANREACH packet to workstation A only. Router A decapsulates the ICANREACH packet to an LLC2 +RSP frame.

LLC2 requires timely responses and acknowledgments between the two end stations. Extending the distance between workstation A and workstation B, and using a slower media than the rated speed of a LAN, might cause LLC2 timers to expire and terminate the connection. Serial WAN link speeds are from 256 Kbps to 2 Mbps, while LAN speed is 100 Mbps.

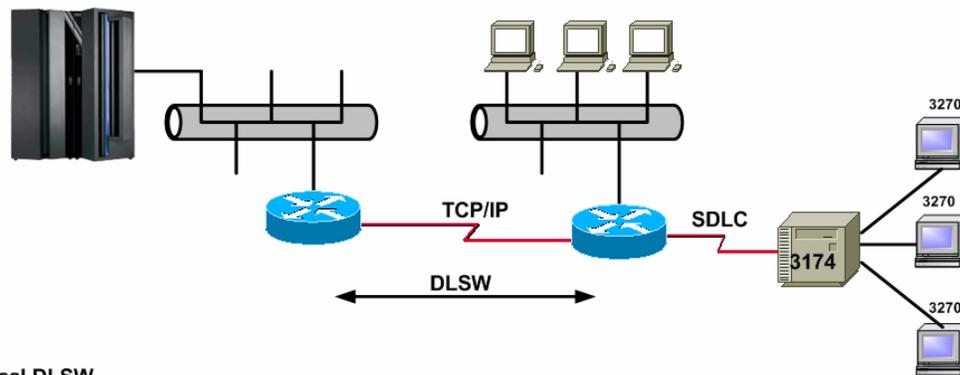
With DLSw, connections are terminated at the DLSw routers, which acknowledge packets locally instead of transmitting these across the WAN. This technique is known as *spoofing*; it reduces WAN traffic, eliminating potential LLC2 time-outs.

## 9.2.4 Connecting SDLC devices using DLSw

DLSw routers were installed in remote branches. The DLSw router enabled the connection of TCP/IP and SNA LAN-attached workstations to the mainframe. Many remote branches still have SDLC devices like IBM 3174 controllers, banking, or retail controllers. Using the DLSw branch router eliminates the need to connect the SDLC device using a dedicated communication link.

Figure 9-3 shows how an SDLC controller in a remote branch is connected using DLSw routers.

A. Remote DLSw



B. Local DLSw

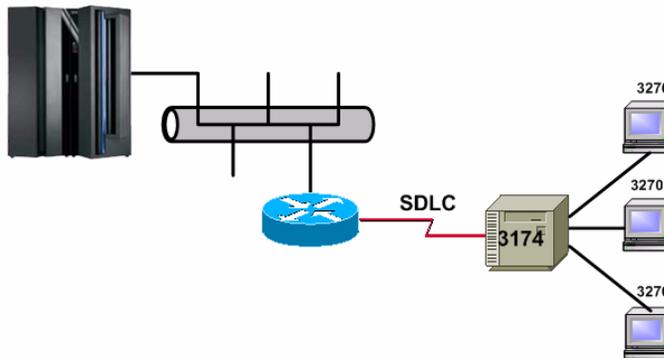


Figure 9-3 Connecting SDLC controller using DLSw

Figure 9-3 depicts the two types of DLSw: remote (A) and local (B).

- ▶ Remote DLSw requires at least two routers (with DLSw feature loaded) connected to each other over an IP network. DLSw performs an encapsulation/decapsulation function: wrapping the SNA frames into IP packets for transportation across the IP network.

- ▶ Local DLSw does not use TCP/IP. Instead, it enables communication between LAN-attached SNA devices and an SDLC device that is link-attached to the same DLSw router.

The Remote DLSw configuration in Figure 9-3 enables the workstations connected to the remote LAN workstations to communicate with either TCP mainframe-based applications using the communication link that connects the two routers, or mainframe SNA application using encapsulated LLC2 in TCP packets (DLSw). The SDLC traffic is converted by the router to LLC2 and is transported using DLSw.

Every router has an IP address assigned to its LAN interface.

Figure 9-4 on page 218 illustrates the representation of the SDLC link inside a DLSw router.

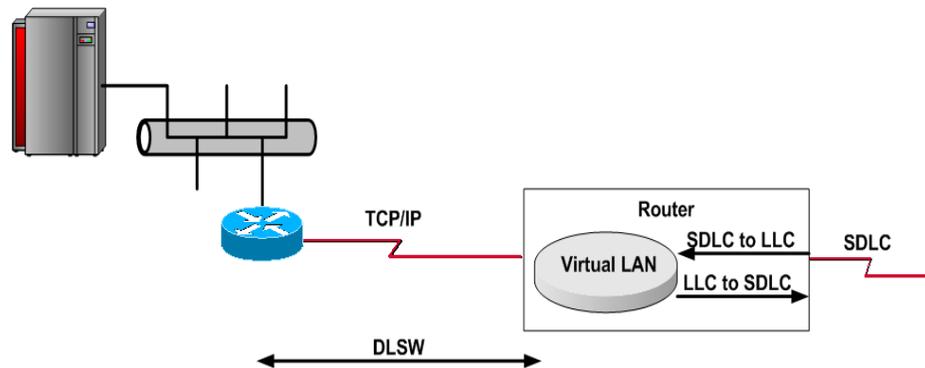


Figure 9-4 SDLC-to-LLC2 conversion

In Figure 9-4, the SDLC link is represented inside the router as a connection to a virtual LAN. Virtual LAN is a LAN implemented inside the router and is not related to a physical LAN interface. The SDLC link is assigned a virtual MAC address, and this MAC address is carried in the encapsulated LLC2 frames.

Figure 9-5 on page 219 shows the placement of DLSw routers in an IP backbone network. DLSw routers are located at the edge of the network adjacent to the SNA device, mainframe, and server in our example. The DLSw routers perform the encapsulation and decapsulation of the LLC frame that carries the SNA path information unit or LLC2 commands and responses.

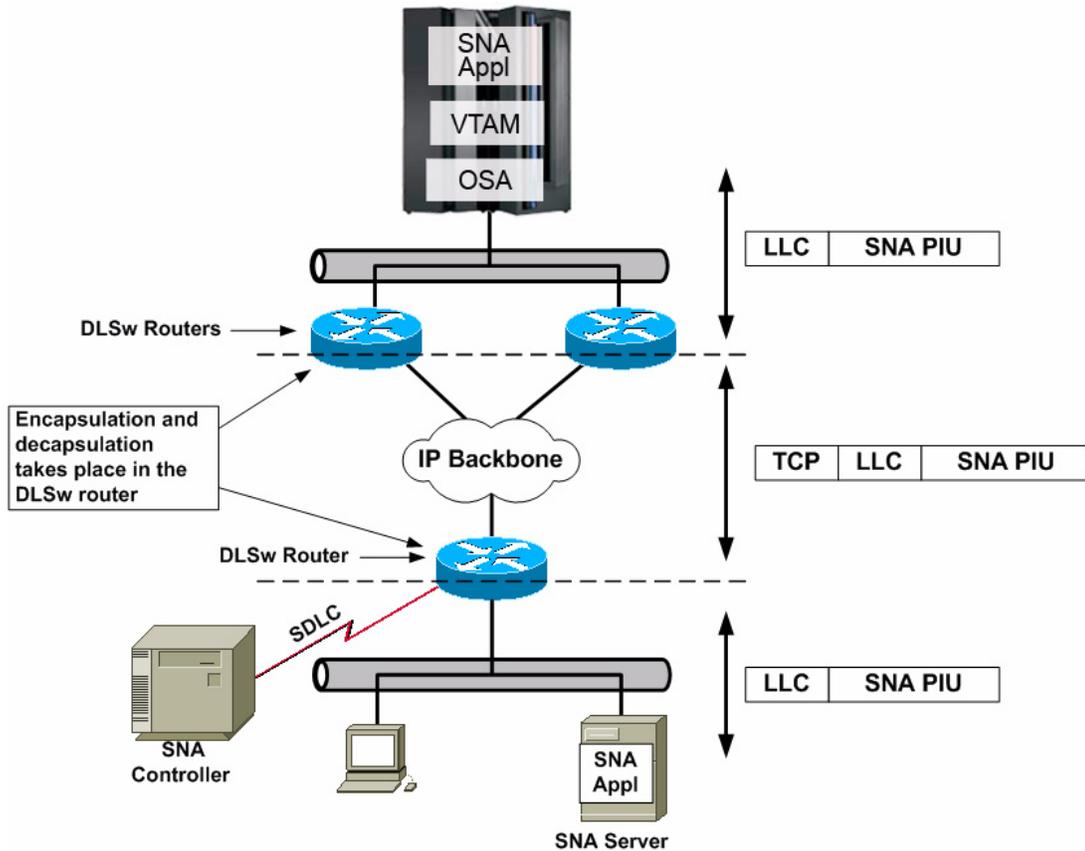


Figure 9-5 A conceptual view of DLSw

The following points summarize some of the DLSw attributes:

- ▶ DLSw is a forwarding mechanism, and supports SNA device types PU2, PU2.1 and PU4.
- ▶ DLSw provides switching at the data link layer (layer 2), and encapsulates SNA data into TCP packets for transport over an IP network.
- ▶ DLSw does not provide full routing capabilities. DLSw, like any other TCP connection, utilizes the dynamic and rerouting capabilities of the IP network.
- ▶ DLSw is a TCP application that uses default ports 2065 and 2066.
- ▶ SNA endpoints remain the same. No change is required to SNA endpoints.
- ▶ Because SNA data is encapsulated in TCP packets, DLSw does not interpret the SNA headers. The consequence is that SNA class of service (COS) is not implemented in the IP network.

- ▶ Some older models of the one gigabit OSA-Express cards in QDIO mode support the IP protocol only. Using DLSw rules out the use of these one gigabit OSA-Express cards and imposes the use of 100 megabit OSA-Express cards. These cards are defined in the HCD as non-QDIO.

DLSw was the first SNA over IP solution that became available. It is a mature product and you will find that many organizations have implemented this solution.

### 9.2.5 DLSw configuration

From a z/OS perspective, the DLSw configuration and definitions reside on the router. As a network administrator, you normally would not get involved with these definitions, because the group responsible for the WAN does them. You need to interact with this group to get MAC addresses of SNA devices and especially the SDLC virtual MAC addresses that the mainframe uses to initiate the LLC2 connection.

For LAN-attached devices, the VTAM definitions remain the same. When changing the SDLC connection from NCP to DLSw router configuration, the physical connection of the devices (PUs and LUs) changes. The router attaches via the LAN, and in most cases will use the OSA card. VTAM definition has to be altered from an NCP definition to a switched major node that includes the physical unit and the logical units of the SDLC-attached device.

## 9.3 Enterprise Extender

Enterprise Extender (EE) has provided a useful solution to the dilemma of running SNA applications over IP networks. “Extending the enterprise” is an appropriate description.

### 9.3.1 What is it?

Enterprise Extender is a standard created by the Internet Engineering Task Force (IETF) and APPN Implementers Workshop (AIW). It is documented in RFC 2353.

The Enterprise Extender architecture carries the SNA high performance routing (HPR) traffic of any logical unit type over an IP infrastructure without requiring changes to that infrastructure. It treats the IP network as a particular type of SNA logical connection. In this manner, the SNA protocols act as transport protocols on top of IP, as does any other transport protocol such as Transmission Control Protocol.

An important aspect of Enterprise Extender is the ability to view the IP network as an APPN connection network (see 8.7, “Connection networks” on page 202). In this case, the benefit comes from the ability to establish dynamically a single one-hop HPR link to any host to which IP connectivity is enabled, provided that the host implements Enterprise Extender. In general, this allows the routing function to be handled entirely within IP. IP routers serve as the only routing nodes (hosts) in the network.

Figure 9-6 on page 222 pictures two backbone networks, an SNA network and an IP network. The SNA network connects SNA devices without encapsulating the data to the mainframe. The IP network connects IP devices like TN3270 clients and TCP clients implementing Web browsers directly to the TCP stack in the mainframe.

Devices that implement Enterprise Extender are located on the border of the IP and SNA network. These devices are connected on one side to the SNA network and on the other side to the IP network. The IP network transports the Enterprise Extender traffic over the IP backbone. The routers inside the IP backbone are pure IP routers not requiring any additional software, as in the case of DLSw.

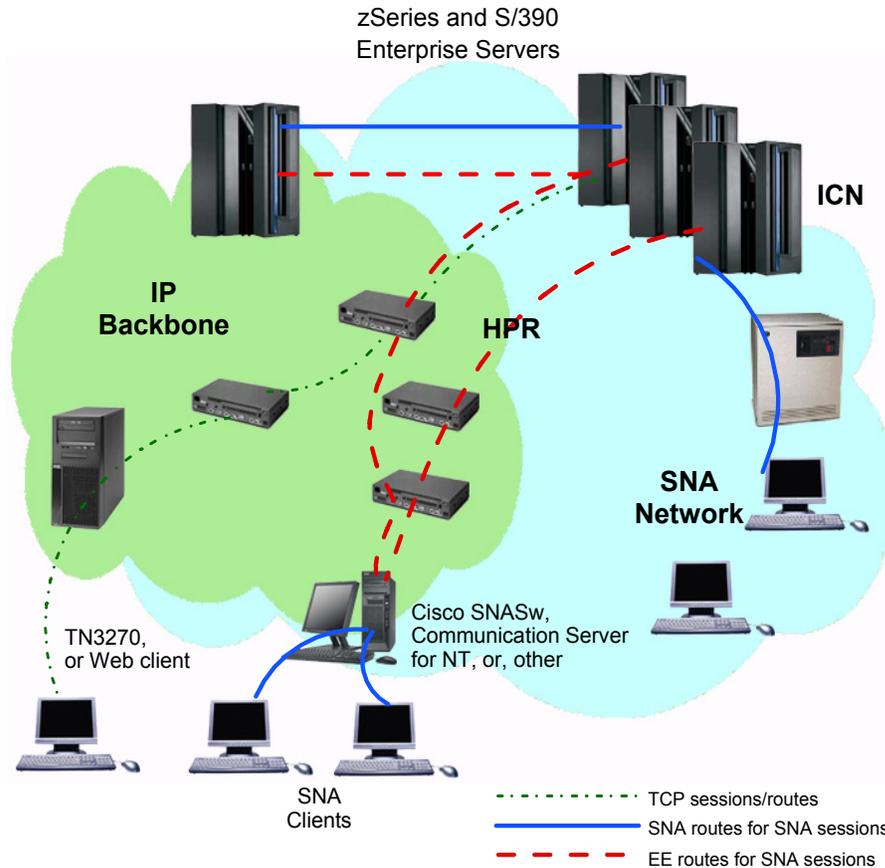


Figure 9-6 TCP and Enterprise Extender backbone network

In the branch side, where the SNA clients exist, special hardware and software converts the SNA packets to IP packets. Figure 9-6 lists either the IBM Communication Server for NT or the Cisco SNA Switching (SNASw), a special version of Cisco's Internetworking Operating System (IOS) that implements branch extender (see "Branch extender (BEX or BrEx or BrNN)" on page 175). The role of SNASw is explained later.

To the SNA HPR network, the IP network appears to be a logical link; to the IP network, the SNA traffic appears as UDP datagrams. The UDP datagrams are routed without any changes to the IP network.

Figure 9-7 on page 223 shows how the two mainframes connected via the IP network cloud. The path that carried the Enterprise Extender IP/UDP datagrams through the IP cloud is the logical link that connects the two mainframes. The

rapid transport protocol (RTP), which is a logical connection, uses the IP network as an HPR link.

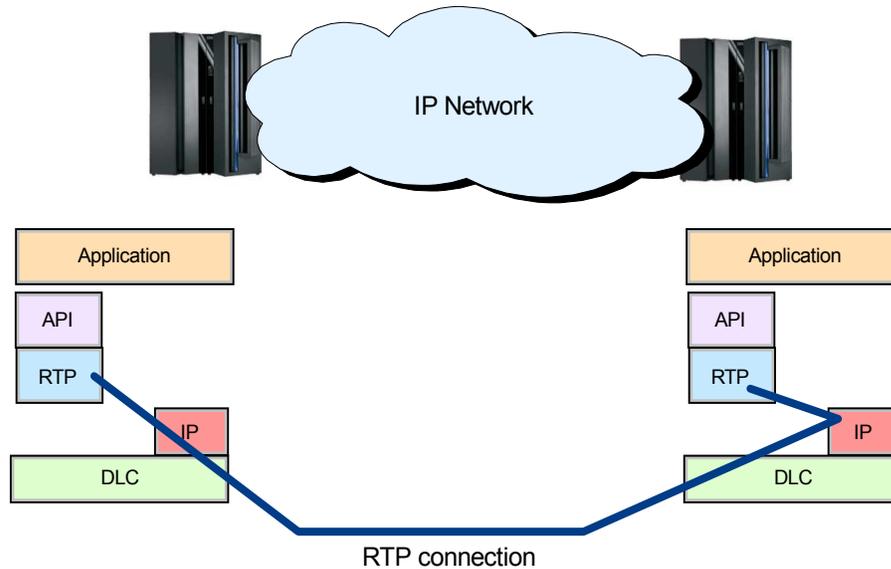


Figure 9-7 SNA HPR over the IP network

In a “mixed” Enterprise Extender and HPR connection, a single RTP pipe connects the two endpoints. The single RTP pipe is made up of two hops:

- ▶ A HPR hop of unspecified type
- ▶ An Enterprise Extender connection

A two-hop mixed HPR and Enterprise Extender is depicted in Figure 9-8.

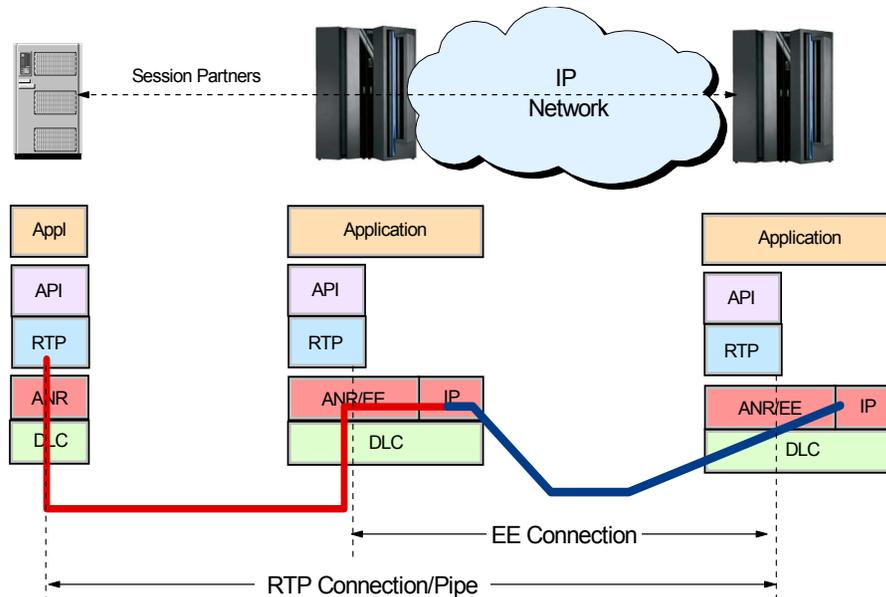


Figure 9-8 Combined HPR and Enterprise Extender connection

Enterprise Extender has been designed to run over existing IP networks without requiring any change to applications or to IP routers. SNA applications see the same SNA network interfaces as before, while IP routers continue to see familiar UDP packets.

### Session availability in mixed EE and HPR

In a mixed EE and HPR connection, as shown in Figure 9-8 on page 224, the rerouting of SNA sessions takes place by the protocol where the failure is identified.

IP has always had the ability to reroute packets around failing components, without disrupting the connection, by means of the connectionless property of IP. More recently HPR has implemented non-disruptive path switching, which provides the same function as an IP network, although in a different fashion.

The HPR extension to SNA is connection-oriented, which has always been a characteristic of SNA. However, when it detects a failure, it moves an existing connection around a failing component. The use of HPR transport over an IP network provides nondisruptive rerouting around failed network components using either IP or HPR methods, depending on the location of the failure.

If the failure occurs in the IP network, the rerouting is handled by the IP network. If the failure is the HPR portion, HPR's non-disruptive path switching reroutes the session to an alternative path.

### 9.3.2 Why does Enterprise Extender use UDP packets?

Thus far we described SNA as a robust and reliable protocol. So why is it that Enterprise Extender transports SNA over IP packets using UDP, which by definition is unreliable and whose transmission is based on best effort?

The designers of Enterprise Extender had the task of architecting the way in which SNA and IP-based protocols would be layered to transport SNA data over the IP network. They had three choices for encapsulating SNA data units: raw IP datagrams, UDP packets, or a TCP connection. Let's take a closer look at each choice in more detail:

- ▶ Raw IP datagrams

Datagrams are completely compatible with the HPR principles because they flow through the network with minimal overhead and provide no error recovery of any sort. However, raw IP provides no means of multiplexing, particularly with no Internet Engineering Task Force (IETF)-designated protocol value for HPR. Using a non-designated protocol value can lead to inconsistencies with security measures that filter IP packets based on this value.

Additionally, although raw IP allows priority and type of service to be specified, in practice not all networks or routers are, or can be, configured to support this.

- ▶ UDP packets

- ▶ These packets provide the multiplexing required because they contain UDP port numbers, which allows Enterprise Extender packets to be distinguished from other IP packets. UDP also permits a priority scheme to be implemented independent of the type of service bits, because many routers can prioritize traffic based on the received port number.

UDP also has low overhead because it does not concern itself with error recovery or flow control.

- ▶ TCP connection

A TCP connection also provides multiplexing through port numbers, but it incurs a significantly higher overhead than raw IP or UDP. A TCP connection handles error recovery, retransmission, and flow control. None of these is required for an HPR connection because the RTP endpoints are responsible for all of them.

## Enterprise Extender on z/OS

The implementation of EE in z/OS involves data transfer between the VTAM and the TCP/IP address spaces. A special connection type called IUTSAMEH is used to move data from VTAM to TCP/IP and vice versa.

This connection type is used to connect two or more Communications Server for z/OS IP stacks running on the same MVS image. In addition, it is used to connect Communications Server for z/OS IP stacks to z/OS VTAM for use by Enterprise Extender.

For Enterprise Extender, z/OS Communications Server implements a separate UDP layer called “Fast UDP” that is optimized for Enterprise Extender communication. Fast UDP, communicates with Enterprise Extender (the APPN over UDP component in VTAM through the IUTSAMEH device.

Figure 9-9 illustrates the Enterprise Extender components in z/OS and the data flow among these components.

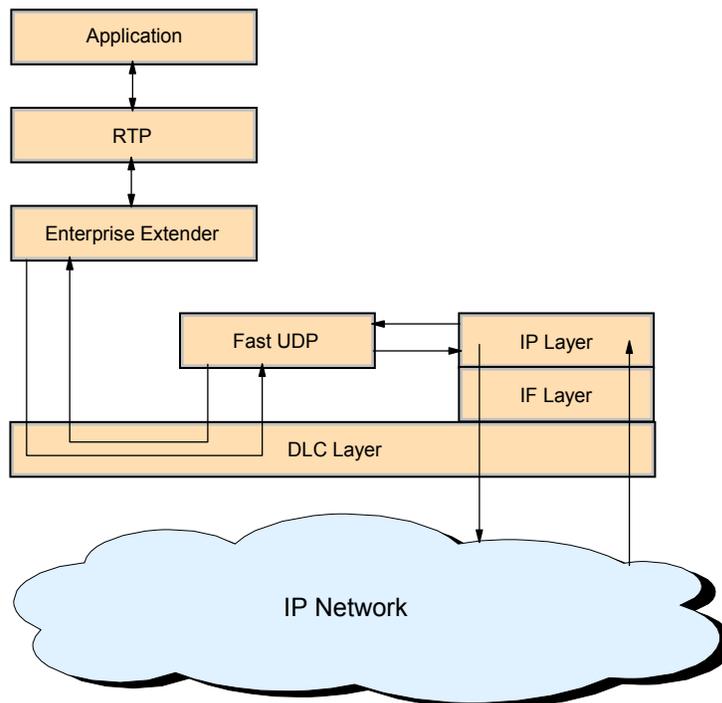


Figure 9-9 Internal Enterprise Extender data flow on z/OS

### 9.3.3 Class of service (CoS)

One of the biggest issues facing those who want to transport SNA over an IP network is the question of maintaining SNA's class of service (CoS). In SNA, the class of service specified for a particular session is used to determine both the route taken by the session and the transmission priority allotted to it.

With an IP backbone, the route is essentially unpredictable because of IP's connectionless property. However, IP provides for a transmission priority using the precedence bits in the IP header. Many routers now support the use of these bits. However, in the past, they tended to use the TCP or UDP port number as a means of assigning priorities to packets.

Enterprise Extender supports the use of both precedence bits and port numbers to inform the IP network of the transmission priority. You should use precedence bits because they are in the IP header. The UDP or TCP port numbers are carried inside the IP datagram, so encrypted packets have unreadable port numbers, and fragmented packets have no port numbers after the first fragment. For such encrypted or fragmented packets, intermediate routers cannot determine the appropriate priority.

#### Using IP type of service (ToS)

The IP header ToS field assigns the priority of the IP packet. Routers along the data path forward packets with higher type of service (ToS) values prior to forwarding lower priority ToS packets.

IP type of service (ToS) is related to the APPN class of service (CoS) and the UDP port numbers used for Enterprise Extender traffic. The APPN CoS specifies a transmission priority (with the keyword PRIORITY), which can be one of the data values listed in Table 9-1, with their corresponding UDP ports.

*Table 9-1 ToS values and corresponding priorities*

SNA priority	IP ToS	UDP port
LLC2 commands	B'110'	12000
Network	B'110'	12001
High (TP=2)	B'100'	12002
Medium (TP=1)	B'010'	12003
Low (TP=0)	B'001'	12004

### 9.3.4 EE implementation in non-z/OS remote sites

There are products and services that support Enterprise Extender on systems other than z/OS.

#### **SNA Switching Services (SNASw)**

The Cisco SNA Switching Services (SNASw) feature supports the Enterprise Extender function. The SNASw function is implemented as a branch network node (BrNN) or branch extender (BX) node. BrNN appears as an APPN network node (NN) to downstream devices (for example, workstations and servers) and an end node to upstream devices (for example, z/OS).

#### **Host Integration Server 2004 and IBM Communications Server for Windows**

Both of these software products provide Enterprise Extender capability directly from an organization's branch or office location to Enterprise Extender on z/OS.

Many organizations have these products deployed for SNA device and application support, connecting to z/OS through DLSw or SNASw routers. New Enterprise Extender support in both products allow the DLSw or SNASw router connections to be eliminated, and an IP WAN router connection is the only requirement.

#### **IBM Personal Communications for Windows, AIX, and Linux (PCOM)**

The IBM Personal Communications family has support for Enterprise Extender. Using the IBM-EEDLC interface configuration option, Enterprise Extender connections can be set up directly from the desktop to Enterprise Extender on z/OS for SNA-based applications. As with Host Integration Server and Communications Server, an IP router is the only requirement.

### 9.3.5 Internet connectivity exploitation

Enterprise Extender enables remote branches or workstations to be connected to the SNA backbone using the Internet, with no application changes required, while maintaining SNA connectivity from end to end. Dependent LU sessions can be carried on an Enterprise Extender connection as easily as any others by using the dependent LU requester function.

Some of the benefits of Enterprise Extender include:

- ▶ The key advantage of being able to consolidate onto a single transport network, which eliminates parallel networks, reduces equipment, lowers data circuit cost, and simplifies network management.
- ▶ There are no changes required to SNA applications.
- ▶ SNA can exploit the OSA Gigabit Ethernet interface cards.
- ▶ End-to-end failure protection and data prioritization using the IP router network and z/OS Communications Server facilities.

## 9.4 Comparing Enterprise Extender and DLSw

Table 9-2 on page 230 compares DLSw and Enterprise Extender.

Table 9-2 Comparison of Enterprise Extender and DLSw

DLSw	Enterprise Extender
DLSw requires significant router resources, in terms of CPU and storage. Data center routers, can be heavily loaded with DLSw connections.	Enterprise Extender on z/OS can reduce the data center router requirements because OSA-Express cards provide the conduit for IP and UDP traffic.
DLSw data center and branch endpoints provide an additional point of failure. Loss of the data center router (providing services for endpoints) results in session outages.	Enterprise Extender has the endpoint in z/OS Communications Server. IP reroutes around a failed data center router.
Does not fully support SNA session priority.	Enterprise Extender maps the SNA session priority to a set of UDP ports and the IP ToS field.
DLSw traverses the full SNA stack and TCP/IP stacks, requiring more resource and time to complete its data transfer.	Enterprise Extender supports only HPR and since HPR provides generally the same functions as TCP (reliable data transfer, error recovery and more), Enterprise Extender can use the unreliable UDP protocol.
Can use only fast ethernet (100 megabit OSA)	Exploits 1 gigabit OSA.
	Throughput and response time comparisons of Enterprise Extender and DLSw show Enterprise Extender outperforms DLSw.
	SNASw, together with branch extender, provides end-to-end Enterprise Extender connectivity using the existing IP WAN infrastructure.

## 9.5 Implementing Enterprise Extender

As already discussed, the motivation to migrate an SNA network to Enterprise Extender is:

- ▶ To carry SNA traffic on the IP network
- ▶ To preserve the investment in SNA applications
- ▶ To connect to business partners using SNA network interconnect (SNI)

The best candidate for Enterprise Extender is an enterprise that has SNA applications in remote branches.

Two topologies that can be considered as candidates for migration to Enterprise Extender are the following:

- ▶ Organizations that still use the older SNA technology with communication lines from the branch to an IBM communication controller
- ▶ Installations that converted their SNA network to DLSw and use an IP backbone that consolidates the IP and SNA protocols

### **Migrating remote branches from the older SNA-based technology to Enterprise Extender**

To migrate from the older SNA technology, an installation must have an SNA-based server in the branch. The SNA software on the server can be one of the following:

- ▶ z/OS Communications Server with Enterprise Extender running on a mainframe host
- ▶ Microsoft Host Integration Server (HIS) 2004
- ▶ IBM Communications Server (CS) for Windows
- ▶ IBM Personal Communications (PCOM) for Windows
- ▶ IBM Communications Server for AIX
- ▶ IBM Communications Server for Linux

You can also use Cisco Routers with the SNA Switching Services feature (SNASw) loaded and configured.

The first step is to decide which platform to use at the edge of the IP backbone and the branch. The two options to implement Enterprise Extender on the SNA-based server or to use Cisco routers with SNA Switching Services (SNASw).

Implementing Enterprise Extender with Cisco routers does not require any additional software or hardware in the branch. If the decision is to implement Enterprise Extender on the server, a vanilla IP router is installed in the branch, and in some cases, an upgrade of the SNA server software is required.

Figure 9-10 on page 232 depicts the Enterprise Extender network.

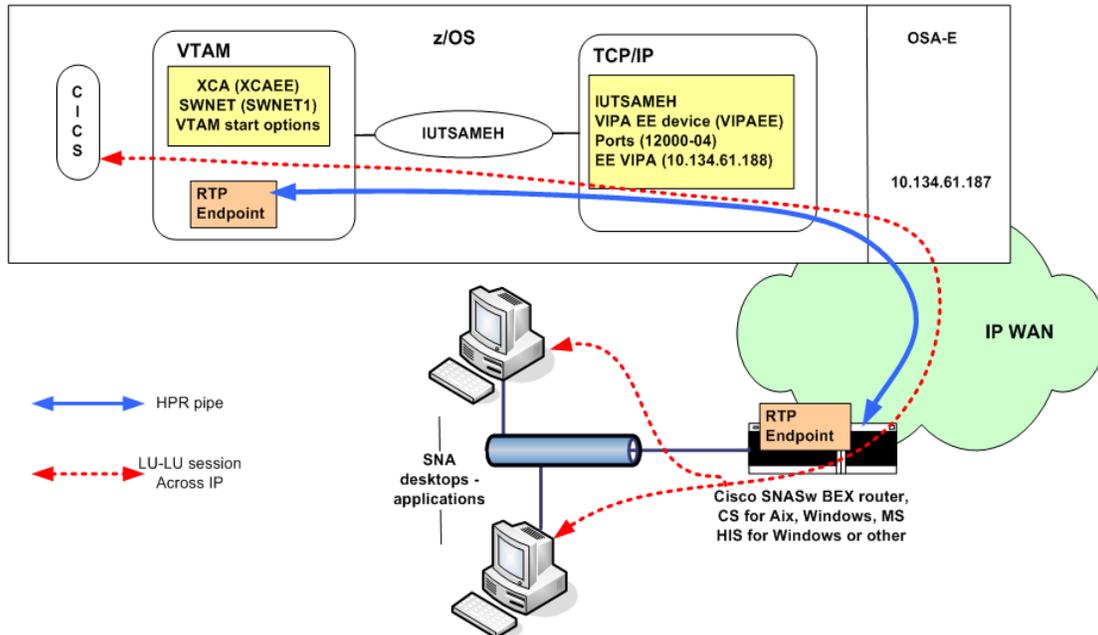


Figure 9-10 Overview of branch extender node connection from branch to z/OS

This figure shows the end-to-end Enterprise Extender flow. The RTP endpoint is within HPR. The HPR packet is integrated into a UDP packet and travels through the IP WAN routers to the data center, through the OSA-E card (which has a unique IP interface address), into the TCP/IP stack Enterprise Extender VIPA, then into the Enterprise Extender VTAM link (IUTSAMEH), and finally to the application.

## Migrating remote branches from a DLSw/IP-based backbone to Enterprise Extender

Despite Enterprise Extender being the preferred solution for transporting SNA data over an IP network, many organizations still have DLSw deployed in the WAN. Typically, you see DLSw used to transport SNA sessions over the WAN, with DLSw and SNASw routers deployed in the host data centers to distribute the SNA session to the host LPAR. The DLSw and SNASw feature code sets may be configured on the same data center router.

It is likely that organizations have a combination of both solutions for a while, because it takes time and money to migrate from DLSw routers to SNASw.

If an application or device is unable to be converted to TCP/IP, reducing the complexity of the environment by implementing SNASw / Enterprise Extender at the branch level is the next best option.

Figure 9-11 on page 233 illustrates a DLSw and Enterprise Extender combined solution.

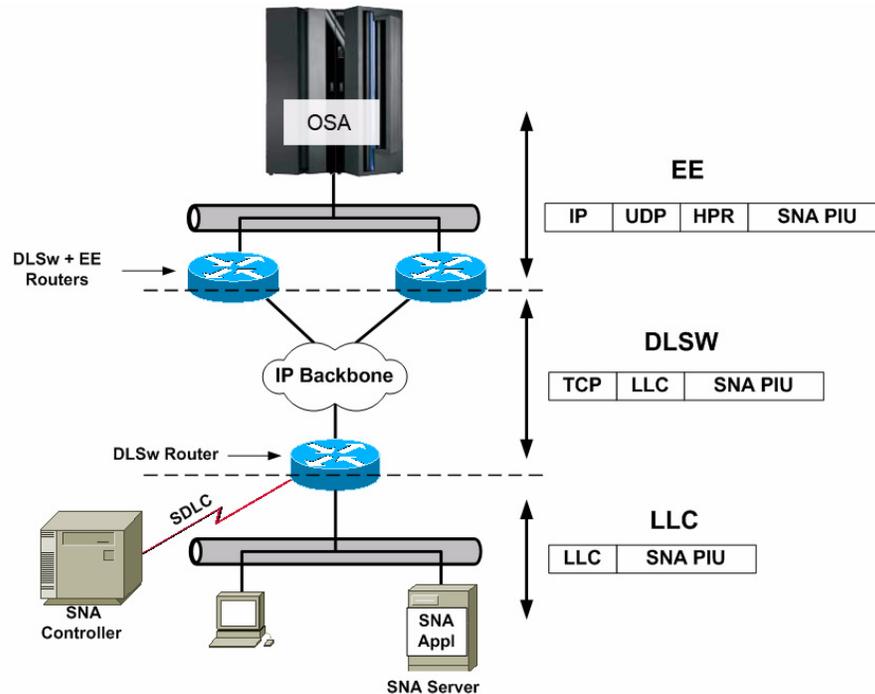


Figure 9-11 Remote branch connection to z/OS using DLSw and Enterprise Extender

## 9.6 Configuration examples

The following examples give you an idea of the SNA/IP implementation parameters you can expect to see defined within the z/OS networking and related components. There are many other environmental and related component parameters that are not shown. For a full understanding of what each parameter does, consult the relevant technical manual.

Figure 9-12 on page 234 illustrates the component areas that the configuration examples define.

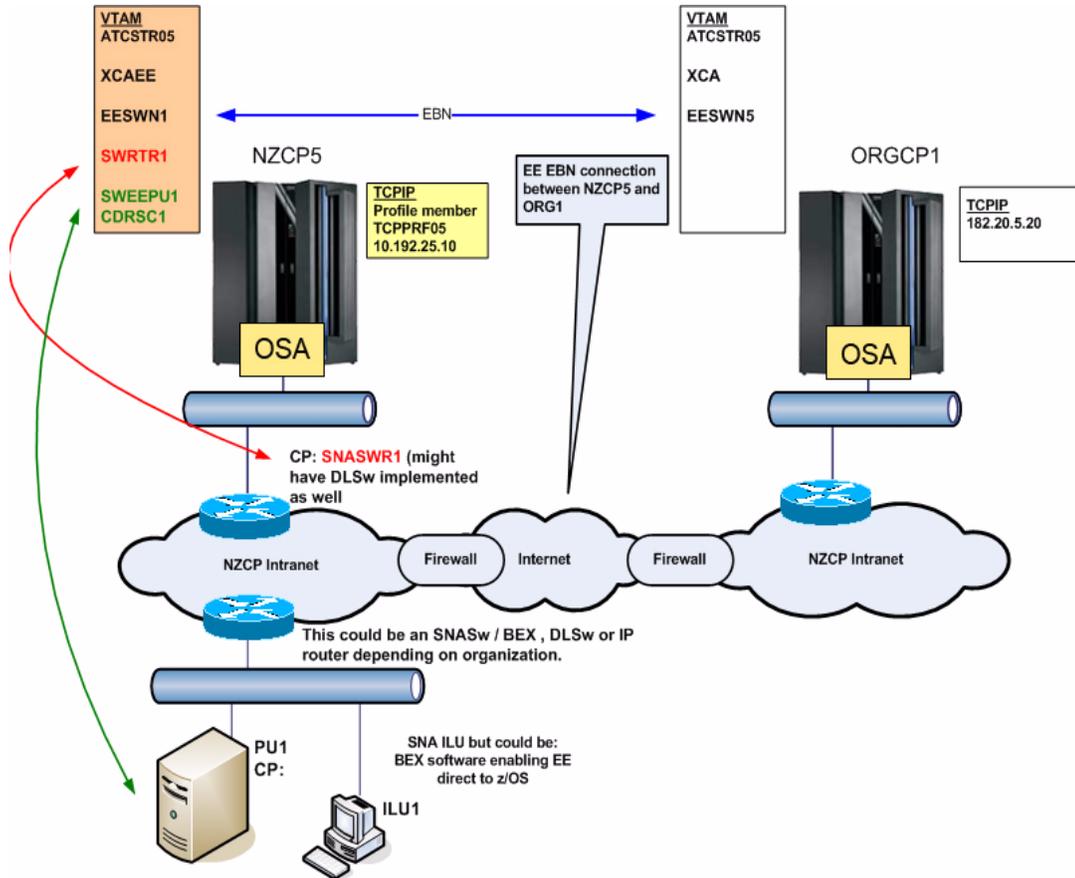


Figure 9-12 Overview of the example environment

## 9.6.1 Enterprise Extender configuration

Enterprise Extender has a number of key parameter definitions that are required in order to enable Enterprise Extender on z/OS. There are also some optional definitions that might be implemented, depending on an organization's requirements.

There are also definitions required on the SNASw routers, and implementing the definitions is normally a job performed by the network group responsible for looking after the WAN equipment. Though this task is usually not handled by a z/OS network administrator, as with any of the data center network equipment, you need to provide some information in order for the SNASw router to connect to z/OS.

A prerequisite of implementing Enterprise Extender on z/OS is that Communications Server must be APPN-enabled.

### **TCP/IP profile statements for Enterprise Extender**

Assuming that TCP/IP address space is up and running prior to the implementation of EE, the following tasks must be performed in the TCP/IP profile definitions:

1. Define a static VIPA and assign it an IP address.
2. Define the IUTSAMEH device.

If, as part of parallel sysplex definitions, you define a distributed VIPA using DYNAMICXCF, then IUTSAMEH is automatically defined.

3. It is recommend that a dynamic routing protocol, like OMPROUTE, be implemented.

If you are using OMPROUTE, there are additional definitions required, but these are not shown.

Example 9-1 illustrates an example of TCP/IP profile statements that are part of the Enterprise Extender definition process. The TCP/IP profile shows device statements for Enterprise Extender, the Enterprise Extender UDP ports, IP address of Enterprise Extender, and start device statement for Enterprise Extender.

The example uses a unique Enterprise Extender VIPA because it provides separation from the IP VIPA and can be monitored independently. Some sites use the IP VIPA as the Enterprise Extender VIPA, as well.

TCP/IP needs a definition for the port represented by the VTAM application (same-host, IUTSAMEH). This must be active before VTAM can establish any Enterprise Extender connections.

#### *Example 9-1 TCP/IP profile*

---

```
;Virtual Device and Link statements
;EE VIPA device
device VIPAEE  virtual 0
link  VLINKEE  virtual 0  VIPAEE

;IP VIPA device
device VIPA1  virtual 0
link  VLINK1  virtual 0  VIPA1

;There will also be other VIPA device statements for the IP VIPA.

;Device statement for OSA device
device OSADEV1 MPCIPA NONROUTER
```

```

LINK  OSALNK1 IPAQENET OSADEV1
;Device statement for VTAM EE connection
device IUTSAMEH MPCPTP
link  EELINK  MPCPTP IUTSAMEH
;
There will also be other device statements for the interfaces such as OSA.

;PORT statements for Enterprise Extender 12000-12004 default ports
12000 UDP NET
12001 UDP NET
12002 UDP NET
12003 UDP NET
12004 UDP NET

;Home Statements
10.134.61.188  VLINKEE      ; This is the EE Static VIPA address
10.134.61.180  VIPA1       ; stack IP VIPA precedes 1st physical device
10.140.40.50   OSALNK1
;Start device statement
start OSADEV1      ;Start real interface devices
start IUTSAMEH    ;Start

```

---

## VTAM statements for Enterprise Extender

Define the Enterprise Extender IP address and TCP name in the VTAM ATCSTRxx configuration list member; see Example 9-2. Do this on each of the LPARs that are Enterprise Extender-capable.

**Important:** We recommend that you use a unique IP VIPA address for Enterprise Extender on each LPAR.

### *Example 9-2 VTAM statements for Enterprise Extender*

---

```

* SYSP.VTAMLST(ATCSTR01)
* APPN definitions (some of which are mentioned below)
APPNCOS=#CONNECT,
CPCDRSC=YES,
CDSERVR=YES,
CPCP=YES,
CONNTYPE=APPN,
HPR=(RTP,NONE)
NN=YES
* EE specific definitions
IPADDR=10.134.61.188
TCPNAME=TCPIP

```

\* External communications adapter (XCA) VTAM major node defines the IP port

```

* connection to the z/OS TCP/IP stack that VTAM will use for EE connections
* SYSP.VTAMLST(XCAEE)
XCAEE1      vbuild type=XCA
EEPORT    port  medium=hrip,
           livtime=10,
           iptos=(20,40,80,C0),
           SAPADDR=4,
           srqretry=3,
           srqtime=15
EEGROUP   group dial=yes
           dynpu=yes,
           DYNPUPFX=EX,
           AUTOGEN=(nnnn,EEXL,EEXP),
           call=inout

* VTAM Switched major node definition member for a remote SNA Server node
* SYSP.VTAMLST(SWEEPUI)
SWPU1  VBUILD  TYPE=SWNET
PU1      PU    MAXDATA=1033,
         ADDR=01,
         CPNAME=SNASRVR,
         PUTYPE=2

* VTAM CDRSC definition member for independant LU (LU6.2) on PU1 above.
* SYSP.VTAMLST(CDRS1)
CDRSC1  VBUILD  TYPE=CDRSC
ILU1      CDRSC  ALSLIST=PU1

* VTAM Switched major node for CISCO SNASw router
* The SNASw router definition examples are contained in the next section.
* SYSP.VTAMLST(SWRTR1)
          VBUILD  TYPE=SWNET
SNASW1  PU      ADDR=01,
           DISCNT=NO,
           DYNLU=YES,
           CPNAME=SNASW1,
           CONNTYPE=APPN,CPCP=YES,HPR=YES,
           PUTYPE=2

```

---

## 9.6.2 Extended border node configuration

Example 9-3 shows the definitions you might have between two interconnected partners. This configuration would replace an SNA network interconnect (SNI) connection between two front-end processors (such as 3745s).

The example includes the type of extended border node implementation definitions you see under VTAM at both organizations. There are most likely firewall definitions, and ports 12000 - 12004 need to be opened.

Consult your security administrator for the firewall definitions.

*Example 9-3 Extended border node configuration*

---

```
* VTAM Start options for EBN in ATCSTR05 member (NZCP5 definition shown)
* ORGCP1 will have similar ATCSTRxx definitions, IPADDR will be: 182.20.5.20
BN=YES
NODETYPE=NN
TCPNAME=TCPIP
IPADDR=10.192.25.10

* VTAM XCA major node required on both NZCP5 and ORGCP1 - shown in Example 9-2
on page 236
* VTAM Switched Major node on NZCP5 for ORGCP1
EESWN1    VBUILD TYPE=SWNET
EE2ORG1   PU      CPCP=YES,
           CONNTYPE=APPN,
           HPR=YES,
           TGP=ESCON,
           DYNLU=YES,
           DISCNT=NO,
           DWACT=YES,
           NETID=ORGNET,
           CPNAME=ORGCP1,
           ISTATUS=ACTIVE
EEPATH    PATH    IPADDR=192.20.5.20, remote TCP/IP address for connection.
           GRPNM=EEGRPIO,SAPADDR=4

* VTAM Switched Major node on ORGCP1 for NZCP5
EESWN5    VBUILD TYPE=SWNET
EE2NZC5   PU      CPCP=YES,
           CONNTYPE=APPN,
           HPR=YES,
           TGP=ESCON,
           DYNLU=YES,
           DISCNT=NO,
           DWACT=YES,
           NETID=NZNET
           CPNAME=NZCP5,
           ISTATUS=ACTIVE
EEPATH    PATH    IPADDR=10.192.25.10, remote TCP/IP address for connection.
           GRPNM=EEGRPIO,SAPADDR=4
```

---

The existing VTAM application definitions remain the same.

### 9.6.3 Cisco SNASw definitions

The Cisco SNASw router configuration is normally done by the WAN network administrator. However, the z/OS network administrator needs to provide input to these definitions.

The minimum required definition will include the following:

- ▶ The IP addresses of the static VIPA that is used for EE
- ▶ The interface in the router that are used for EE.
- ▶ The location (NETID and CPNAME) of the primary and backup DLUS
- ▶ All VTAM logmode and COSNAMEs names in use

Not all the required definitions are shown in Example 9-4, but this gives you an idea of what type of parameters are set for Enterprise Extender.

*Example 9-4 Cisco SNASw definitions*

---

```
snasw rtp pathswitch-timers
snasw cpname NETX.SNASWR1 ==> network name of router
snasw dlus NETX.SSCP1 backup NETX.SSCP2 ==> Primary and backup DLUS server
snasw port xxxx hpr-ip xxxxx ==> Lan interfaces and connection types
snasw link SSCP1 port EE ip-dest 10.134.61.188 nns ==> define uplinks/lpars
snasw link SSCP2 port EE ip-dest 10.134.61.189 nns
snasw link xxxxx port EE ip-dest 10.134.61.190
snasw mode INTERACT cos #INTER ==> code all VTAM Logmode/COS entry names
```

---

## 9.7 Summary

Consolidating SNA onto IP is not a simple task. Each set of SNA components and applications should be evaluated on its own merits, and a solution should be found that best suits the requirements. In some cases a solution is to replace an application or device; in other cases it is to use DLSw, SNASw, BEX, Enterprise Extender, or a combination of these.

DLSw is a common SNA over IP solution found in the wide area networks (WANs) of many organizations. The DLSw router is deployed at branch offices and peered to data center DLSw routers. Upstream to the z/OS mainframe is normally through SNASw routers and OSA-Express. DLSw provides switching at the data link layer (layer 2), and encapsulates SNA into TCP packets for transportation through the WAN.

Enterprise Extender is an extension of SNA high performance routing (HPR) and provides encapsulation of SNA packets into UDP frames at the edges of an IP network.

Enterprise Extender capable devices and components include: z/OS Communications Server, Cisco routers, Microsoft Host Integration Server and IBM Communications Server for Windows and AIX, and IBM Personal Communications for Windows.

SNASw branch extender support within the above components can be deployed at the branch office, and connects directly to Enterprise Extender on z/OS through an organization's IP router network.

SNA views Enterprise Extender as just another data link control, while IP views Enterprise Extender as a UDP application. No changes are required to the SNA applications if Enterprise Extender is deployed.

Core z/OS enablement definitions are contained in VTAM, but also requires TCP/IP setup. A routing protocol such as OSPF, although it has no Enterprise Extender-specific definitions, is required to provide a high availability IP routing solution.

<b>Key terms in this chapter</b>		
APPN Implementers Workshop (AIW)	automatic network routing (ANR)	branch extender (BX, BrNN, BEX)
data link control (DLC)	data link control (DLC)	IBM Personal Communications (PCOM)
Internet Engineering Task Force (IETF)	rapid transport protocol (RTP)	SDLC
SNA Interconnect	SNA Switching Services (SNASw)	spoofing
Telnet 3270 (TN3270, TN3270E)	Token Ring	wide area network (WAN)



## TN3270 Enhanced

**Objective:** After completing this chapter, you will be able to:

- ▶ Identify the fundamental characteristics of the 3270 data stream.
- ▶ Understand the purpose behind the TN3270 Enhanced (TN3270E) protocol.
- ▶ Understand the functional flow of a TN3270E session.
- ▶ Understand the basic configuration elements of the TN3270E server.

## 10.1 Introduction

During the last several decades, before the Internet Protocol's rise in popularity, large organizations established their own SNA networks. These SNA networks were used to communicate between remote end-users and the centralized mainframe. The display management protocol used to facilitate this communication within an SNA environment was called the 3270 data stream. At the end user's location in an SNA network was a device referred to as a 3270 terminal.

A 3270 terminal was a non-programmable (sometimes called “dumb”) workstation. Stated more simply, it was a display screen with a keyboard attached; see Figure 10-1 on page 243. The 3270 terminal had only rudimentary communications capabilities and was text-based. One of the earliest model 3270 terminal displays (3278 model 1) consisted of 12 rows and 80 columns of text characters, no more and no less. Eventually, a 24 x 80 screen size became the standard, with some alternate sizes available.

To give the old 3270 terminal credit, it did support a selector pen and even a magnetic strip reader. The selector pen was light-based (optical) and it was used to select options on the text screen, similar to how a mouse is used—but of course, the 3270 terminal did not support a mouse.

The 3270 terminal, containing a non-programmable display and keyboard (see Figure 10-1 on page 243), was usually connected to a control unit using coaxial cable (although over time other connection choices became available). The device type for these control units, up until the time the Internet Protocol began displacing SNA networks, was called a 3174 control unit.

The 3174 had some programmability, which allowed an expansion of the capabilities and connectivity options of a 3270 workstation, but it still was a long way from today's GUI workstations. 3270 display terminals were attached to the 3174 using ports, with up to 64 terminals capable of connecting to a single 3174. The 3174 control unit had other capabilities as well, including printer support. At the time of the 3174's demise, it had even expanded to include support for Ethernet LAN.

During the most recent decade, corporate networks started implementing IP as the transport protocol on their backbone. Because so many SNA and 3270 applications existed, they looked at integrating the SNA protocol into their IP backbone. The technology used to move from SNA 3270 applications to TCP/IP is called TN3270, short for Telnet 3270.



*Figure 10-1 IBM 3270 Display Terminal*

### **10.1.1 3270 data stream**

Prior to explaining the TN3270 implementation, you need to know a little more about the 3270 data stream.

The 3270 data stream operations are designed primarily for transmitting data between an application program and a 3270 display with keyboard so end users can interact with mainframe-based applications.

The 3270 data stream is also designed for transmission of data to 3270 printers.

The 3270 data stream is implemented using a mapped character buffer in the device. In turn, this mapped character buffer forms the display on the screen. Data received from the application program and data to be transmitted to the application program are stored in a device buffer and displayed on the screen in the form of alphanumeric characters and symbols. The displayed data is updated when the end user modifies the buffer data and when new data is received from the application program. Each character storage location in the buffer maps to a character position on the display.

All characters and numbers are represented through Extended Binary Coded Decimal Interchange Code (EBCDIC).

### **Write control character (WCC)**

The 3270 write control character is often included in the 3270 data stream. It is used to sound a beep or to unlock the keyboard.

### **Data stream commands**

The 3270 data stream commands perform various functions. The basic 3270 data stream commands are:

- Write** Write characters to the terminal display.
- Erase/Write** Erase and write characters to the terminal display.
- Erase All Unprotected** Erase all unprotected fields on the terminal display.
- Read Modified** Read unprotected data fields that have been changed on the terminal display.
- Attention Identifier (AID)** Send control information (function keys, attention key) to the mainframe. The AID is end-user initiated.
- Write Structured Field (WSF)** Send control information (color, number of rows and columns at terminal, for example) to the terminal display.

**Function and attention keys:** Function keys are the numbered keys prefixed by the letter F found across the top of a standard keyboard. Function keys are heavily used in the TN3270 environment as shortcuts for interacting with applications. For example, F7 and F8 are often page up and page down. A function key is analogous to a shortcut, such as using the Alt + Tab keys to switch applications in a Windows environment.

The attention key is used to present an external interruption to an executing application. Attention interruptions are used to halt execution.

While these commands appear straightforward, what is the meaning of, say, an unprotected field? All locations on a 3270 display screen are governed by field attributes.

### Field attributes

The field attribute defines the start of a field and the characteristics of the field. Here are some common field attributes.

**Protected or unprotected.** A protected field cannot be modified by the end user. The end user can enter data or modify the contents of an unprotected field. Unprotected fields are classified as input fields.

**Nondisplay or display.** Nondisplay means that any characters entered from the keyboard are placed into the buffer for possible subsequent transmission to the application program, but they are not displayed. This is often used for password entry fields.

**Intensified display.** Intensified display means the characters appear on the screen brighter than other (un-intensified) characters.

**Alphanumeric or numeric.** Unprotected alphanumeric fields are fields into which an end user normally enters data using the shift keys (uppercase/lowercase or numeric/alphabetic). A numeric field can contain only numeric characters, with the addition of the period (to indicate a decimal) and minus sign (to indicate a negative number).

Field attributes can be further modified using extended field attributes. Extended field attributes allow such characteristics as foreground and background colors and highlighting to be controlled.

These 3270 data stream fields appear reasonably straightforward, but the implementation details can quickly become very complex; *3270 Data Stream Programmer's Reference* provides exhaustive details.

## 10.2 TN3270 Enhanced (TN3270E)

Today, a single instance of the TN3270E server can support up to 128 000 emulated 3270 display terminals. Display terminals are emulated in software called *TN3270E clients*, which can run on a standard personal computer or workstation. In the world of genuine 3270 display terminals, 128,000 end users would require an enormous amount of dedicated, limited function keyboard and display devices, not to mention 2000 dedicated 3174 control units! As network usage exploded, the SNA 3270 method of communicating with a mainframe became untenable. The solution came in the form of the Internet Protocol (IP).

### 10.2.1 Telnet protocol and SNA meet

The original basic telnet protocol was defined in RFC 854. This RFC effectively defined all that was needed to support the 3270 data stream, since the 3270 data stream is just part of the telnet data payload.

In other words, the 3270 portion was implemented outside of (above) the telnet protocol. Specific options could be negotiated (beyond basic telnet) using the telnet option standard of RFC 855. Option negotiation in turn allowed for device type negotiations (later formally defined in RFC 1091) to be completed as part of the telnet session setup.

Initially, there was no formal standard for TN3270 (the E came along later), but it was clarified in an early RFC titled “TN3270 Current Practices” (RFC 1576). TN3270 itself began to take shape more formally with RFCs 1646 and 1647. RFC 1647 was a significant RFC because it was the first formalization of the TN3270 Enhanced protocol, known as TN3270E.

TN3270E improved upon the TN3270 protocol to include control of LU name selection, as well as full support of Attention Identifier such as SYSREQ (to talk directly to VTAM) and the attention key.

**Reminder:** LU stands for logical unit. An LU is the SNA entity that represents an endpoint of communication for a session. For example, an LU can represent an application endpoint (for example, TSO on the mainframe) or an end-user endpoint (a user at a workstation). The end-user endpoint LU is referred to as the *terminal LU*.

In addition, if an SNA application wants to send data to a printer, it will form a session between the application and an LU that represents the printer.

Today, TN3270E is the standard IP-based method of communicating with a mainframe. Often, the term TN3270 is used synonymously with TN3270E, since

from an end-user perspective it can be difficult to tell the difference. Most TN3270 clients, however, run TN3270E, even though end users may think they are using the TN3270 protocol.

TN3270E is currently defined in RFC 2355.

## 10.2.2 TN3270E description

A TN3270E client uses the TN3270E protocol to access the resources on a TN3270E server. However, the TN3270E client cannot complete the connection all the way to the target application because the TN3270E client communicates according to the TN3270E protocol, while the target application expects communication to be SNA protocol.

Figure 10-2 shows how the communication ultimately reaches the target application (in this example, TSO—but any SNA application can be targeted).

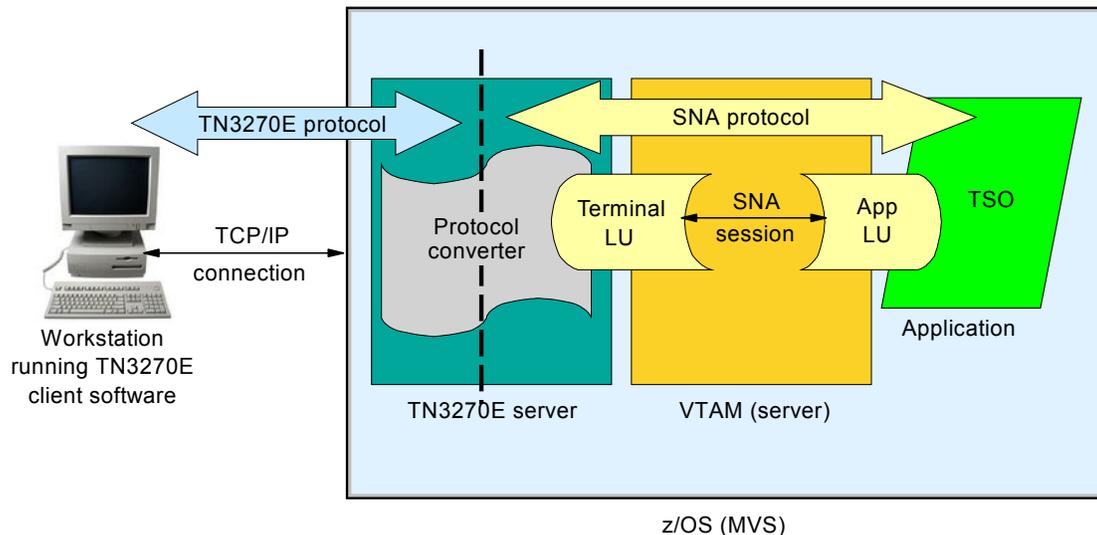


Figure 10-2 TN3270E protocol connection

In effect, the TN3270E server is nothing more than a protocol converter: on one side it maintains an RFC 2355-compliant TN3270E session; on the other side, it emulates a 3270 data stream terminal (including the 3174 control unit) to VTAM. The target application cannot tell the difference between a genuine 3270-attached terminal and a TN3270E server-emulated terminal.

**Note:** The TN3270E server is primarily a protocol converter (from a TN3270E to SNA and back again). However, the TN3270E server does intercept and alter some 3270 data stream flows. It can also alter the sequence of SNA flows. Consequently, it is more than just a converter.

Among the many capabilities of TN3270E, one is the ability of the TN3270E client specifically to request an LU to be used as the terminal LU. This level of control allows more control, from an SNA point of view.

## Session setup

The characteristics of the TN3270E connection are negotiated during the start of the connection. For example, the client can request an LU name, a 3270 function can be negotiated, and, perhaps most significantly, the client must choose a 3270 device type.

**Device types:** There was more than one model of 3270 display terminal. IBM 3278 models 1 through 5 were the most common 3270 display terminals. TN3270E supports models beginning with 3278-2 through 3279-5.

TN3270E also supports the IBM-3287-1 printer device type.

The primary significance of device type selection is the number of rows and columns available in the TN3270E client window.

## TN3270E printing

The TN3270E server has more than just 3270 terminal display capabilities. It can also support the SNA print data stream. Using the same scenario as in Figure 10-2 on page 247, but substituting “printer LU” for “terminal LU” through some SNA session-initiation differences, a print data stream can be emulated by the TN3270E server.

This means that an SNA application can direct a print data stream to an SNA printer LU as it always has. If that printer LU is a TN3270E-owned printer LU, then the TN3270E server accepts the print data stream from the application and forwards it to the TN3270E client running on the workstation. The workstation can then print the data using normal workstation printer facilities.

TN3270E allows the terminal session LU to identify implicitly the printer session LU that should be used. This is done using the ASSOCIATE command during the TN3270E printer connection setup. If an ASSOCIATE command is sent, then the TN3270E server selects a printer LU based upon the terminal LU to which the client is already connected. Obviously, the TN3270E server must be set up with a one-to-one mapping of terminal LUs-to-printer LUs.

## 10.2.3 Additional functions supported in z/OS

Here are some TN3270-related functions that z/OS Communications Server supports:

### Transport Layer Security (TLS)

z/OS Communications Server supports TLS (and of course its older sibling, Secure Sockets Layer, SSL). TLS provides secure data transmission between the TN3270E Server and an TLS-capable client. In an TLS session, any data on a secure port is protected using one of several optional cipher suites. Note that since TN3270E protocol ends at the TN3270E server, data travelling over the SNA session is not protected.

Client authentication and SAF level checks are supported.

**Reminder:** SAF stands for System Authorization Facility. SAF allows mainframe applications to use an external security manager to control access to resources, such as data sets and MVS commands. SAF also forms a user ID repository which can be used to authenticate a TLS client.

### Support for multiple ports

The TN3270E server can listen on multiple ports. In addition, more than one instance of the TN3270E server can run concurrently. Finally, within a single TN3270E server, or among separate TN3270E servers, listening can be controlled so that it is only active on one specific IP address.

Through these functions you can define different security levels (basic or secure) or different configuration parameters, or both, for each port and IP address combination.

### Mapping an IP address to an LU name (IP filtering)

This function provides the ability to select both an LU name and an application name for incoming TN3270E sessions. The selection is made on the basis of a specific IP address, a group of IP addresses, a subnet, or the link name used to connect to the z/OS host. The function makes the LU name and the application name predictable and controllable.

In addition, z/OS Communications Server supports selections based on an IP host name or group of names, as well as an IP address. With the increasing use of dynamic IP (where a given client is not tied to a specific IP address), this can be beneficial in maintaining control over the mapping.

## 10.3 Positioning a TN3270E server

In Figure 10-2 on page 247, note that the connection from the TN3270E server to the target application is “normal” SNA protocol and, because of this, the target application could actually be anywhere within the SNA network—on another mainframe host in the organization, or even in another organization.

You can also place the TN3270E server on a host other than the mainframe. There are TN3270E servers available on the following platforms:

- ▶ AIX with IBM's Communications Server
- ▶ Windows with IBM's Communications Server
- ▶ Windows with Microsoft's Host Integration Server

Other TN3270E software packages are available.

Though a TN3270E server can run in many places, there are reasons to implement it on the mainframe:

- ▶ If all your 3270 applications are in the same place (same host or same Parallel Sysplex), then it makes sense to implement TN3270E in z/OS Communications Server.
- ▶ Through the judicious use of the workload manager and the DNS-based functions of z/OS Communications Server, you can create a resilient TN3270E server that can handle tens of thousands of connections.
- ▶ If all your 3270 applications are in the same place, there is no requirement for SNA in the remote locations or in the backbone.
- ▶ Even if your 3270 applications are in the same location but on different LPARs, you can still implement TN3270E in the mainframe and use a channel-to-channel connection between the LPARs to connect the TN3270E server to the target application.

## 10.4 Implementing the TN3270E server

You can implement TN3270 in z/OS Communications Server either within the TCP/IP address space or, beginning with z/OS V1R6, in its own address space. The definitions of the TN3270E server are identical in both configuration alternatives. However, when running the TN3270E server as its own task, the definitions for the TN3270E server cannot be placed in the TCP/IP profile data set.

The startup JCL has a PROFILE DD statement that points to a profile data set that contains parameters to control the TN3270E server. Within this profile data

set are two fundamental statements blocks used to define TN3270E server behavior:

- ▶ TELNETPARMS
- ▶ BEGINVTAM

A discussion of significant statements from within these statement blocks follows.

### 10.4.1 TELNETPARMS statement block

The TELNETPARMS statement block (ended with ENDTELNETPARMS) contains the TN3270E protocol and other non-VTAM attributes. Example 10-1 shows a sample TELNETPARMS block statement.

*Example 10-1 TELNETPARMS block example*

---

```
TELNETPARMS
  DEBUG EXCEPTION CONSOLE
  TCPIPJOBNAME TCPIP
  PORT 23
  INACTIVE 14400
  PRTINACTIVE 14400
  TIMEMARK 600
  SCANINTERVAL 300
  LUSESSIONPEND
  MSG07
  TELNETDEVICE 3278-2-E NSX32702,SNX32702
  TELNETDEVICE 3278-3-E NSX32703,SNX32703
  TELNETDEVICE 3278-4-E NSX32704,SNX32704
  TELNETDEVICE 3278-5-E NSX32705,SNX32705
  TELNETDEVICE 3287-1          ,SNX32702
ENDTELNETPARMS
```

---

The following provides information about basic TN3270E server setup. You can find complete details on TELNETPARMS block statements in *z/OS Communications Server IP Configuration Reference*.

#### **DEBUG EXCEPTION CONSOLE**

Use this statement to activate TN3270E session tracing. When EXCEPTION is specified, only time-outs and explicit errors are reported.

Many options are allowed on the DEBUG statement. The most commonly used is the DETAIL option. The DETAIL option should be used only when debugging a specific problem and under a controlled environment because the option can produce a large amount of output. The benefit is that the option can quickly point out where a problem lies.

The last parameter on the DEBUG statement controls the destination of where trace records are directed. If trace records are to be written to the JOBLOG instead of the z/OS console, the CONSOLE parameter can be changed to JOBLOG. If no destination is specified (that is, if this last parameter is not coded), the destination default is JOBLOG.

### **PORT 23**

This statement tells the TN3270E server that this block of TELNETPARMS statements applies to connections using PORT 23. This statement also causes the TN3270E server to listen on port 23.

The PORT statement can include an IP address to associate the port with a specific IP address.

### **INACTIVE and PRTINACTIVE 14400**

The purpose of these statements is to clean up sessions that are simply inactive. If a terminal or printer SNA session is inactive for 4 hours, then the TN3270E server ends both the TN3270E connection and the SNA session. Note that the activity only applies to communication occurring on the SNA session side of the TN3270E server (see the right-hand side of the TN3270E server in Figure 10-2 on page 247).

### **TIMEMARK 600 and SCANINTERVAL 300**

These two statements work in tandem and apply to the TN3270E protocol side (the left-hand side of the TN3270E server in Figure 10-2 on page 247).

The TIMEMARK statement controls how often the TN3270E server should send out an “Are you there?” packet to a TN3270E client that has not had any activity. After 600 seconds of idle time, this TN3270E server sends a DO TIMEMARK packet to the client. If any kind of a response is received, the connection is marked as healthy.

The SCANINTERVAL statement controls how frequently the TN3270E server should be checking for healthy connections. For this configuration, every 300 seconds the TN3270E server checks to see if any connections have been idle for more than 600 seconds. The TN3270E server sends the clients of such connections a DO TIMEMARK packet. If any such connections have not sent out a response by the time the next SCANINTERVAL is performed, then the SNA and TN3270E portions of the session are both cleaned up.

### **LUSESSIONPEND**

When a client user enters the LOGOFF command to end a session with a VTAM application, the LUSESSIONPEND prevents the connection from being dropped.

Instead, the client is returned to the screen from which the logon originally was invoked.

**Note:** There are three different places that an initial TN3270E connection can be directed to: the USS (Unformatted System Services) message 10 screen, the default application, or the telnet solicitor screen. More information on this subject can be found in 10.4.2, “BEGINVTAM statement block” on page 253.

### **MSG07**

This statement enables a USS message table to send a USS message 7 to the client in the event of a logon failure. This statement should normally be coded. Message 7 is the number assigned to a the USS message for logon failures. More on USS messages is found in 10.4.2, “BEGINVTAM statement block” on page 253.

### **TELNETDEVICE statements**

As mentioned earlier, the TN3270E session setup requires that the client identify what type of TN3270 terminal it wants to represent.

The TELNETDEVICE statement is used to control the SNA session characteristics that are to be used for both the TN3270E portion and SNA portion of the connection. The second column indicates the device name. The last two columns indicate the logon mode entry (session characteristics) to be used for a TN3270 and a TN3270E connection, respectively.

There are many more configuration statements, which are not shown here, relating to the TELNETPARMS block.

## **10.4.2 BEGINVTAM statement block**

The BEGINVTAM statement block (ended with ENDVTAM) is used to define characteristics that are related to the mapping of the VTAM configuration. A sample BEGINVTAM statement block is in Example 10-2 on page 254.

*Example 10-2 BEGINVTAM block example*

---

```
BEGINVTAM
  PORT 23
  DEFAULTLUS TCP00001..TCP00099          ENDDFAULTLUS
  LUGROUP LUGRP1 LUT101..LUT400..FFFXXX ENDLUGROUP
  PRTGROUP PRTGRP1 PRT101..PRT400..FFFXXX ENDPRTGROUP
  IPGROUP IPGPAY 255.255.0.0:9.8.0.0 ENDIPGROUP
  LUMAP LUGRP1 IPGPAY
  PRTMAP PRTGRP1 IPGPAY
  USSTCP USSTELT
; DEFAULTAPPL TSO
  LINEMODEAPPL TSO
  ALLOWAPPL TSO* DISCONNECTABLE
ENDVTAM
```

---

The capabilities of mapping within a BEGINVTAM statement block are complex and extensive. The sample used here has been kept simple to facilitate explanation of the basic concepts.

### **PORT 23**

This statement is used to connect this BEGINVTAM block with a TELNETPARMS statement for the same port number. In other words, a connection to port 23 uses these BEGINVTAM statements as well as the TELNETPARMS statements for the same port number.

### **DEFAULTLUS**

When a TN3270E client connects to the TN3270E server, it needs to be mapped to an LU that the TN3270E server can use to represent this client on the SNA session. If the client does not specify a specific LU, and if no other mapping statement directs a different LU to be used for this client, then an LU from this default mapping is assigned to the connection.

In this sample, the LU range from TCP00001 to TCP00099 are available as default LUs.

### **LUGROUP**

This statement does not do any mapping. Instead, it defines a group of LUs that can be used for TN3270E terminal sessions. The FFFXXX pattern indicates that the first three characters (for example, LUT) are fixed while the remaining four characters represent a hexadecimal range.

## **PRTGROUP**

This statement does not do any mapping. Instead, it defines a group of LUs that can be used for TN3270E printer sessions. The FFFXXX pattern indicates that the first three characters (for example, PRT) are fixed while the remaining four characters represent a hexadecimal range.

## **IPGROUP**

Again, this statement does not do any mapping. It defines a group of IP addresses (specifically, IP addresses with a 9.8 network ID) that identify TN3270E clients (an IPGROUP is referred to as a client identifier, and there are many other client identifiers available).

## **LUMAP**

This statement maps the LUGRP1 group of LUs to the IPGPAY group of clients. In other words, connections from the 9.8 network can only use terminal LUs from within the LUGRP1 range.

## **PRTMAP**

This statement maps the PRTGRP1 to the same network, such that a TN3270E connection from a client on the 9.8 network can associate a printer from this group.

**Note:** Together, these two LUMAP and PRTMAP statements form the one-to-one mapping that is necessary for a TN3270E printer connection to utilize the ASSOCIATE command.

## **USSTCP**

This statement specifies that the USS message 10 panel (a logon panel similar to native SNA terminals) be presented for the initial connection. If LUSESSIONPEND is coded, a client is returned to this screen after logging off from an application.

**Note:** Unformatted System Services (USS) messages are messages used in an SNA environment to facilitate application access. USS message 10 (often referenced as USSMSG10) is the standard logon message presented when a session is initially established. USSMSG7, as mentioned, is the standard error message presented when a command entered at a USSMSG10 screen fails to complete successfully.

## **DEFAULTAPPL**

Note that this statement is commented out. That is because a DEFAULTAPPL and a USSTCP statement have the same function: they control where a user is directed at initial connection time. If a USSMSG10 panel is not desired, this statement could be used to direct a client to a specific application (TSO for example) at connect time.

## **LINEMODEAPPL**

It is possible for a client to negotiate line mode when connecting to the TN3270E server. In such a situation, this setup connects the client to the TSO application.

## **ALLOWAPPL**

This statement limits the selection of application for a TN3270E client to the TSO application only. This is a security statement to control what applications can be selected from the USSMSG10 panel.

### **10.4.3 VTAM setup for TN3270E server**

If the TN3270E server is going to acquire LUs on behalf of a TN3270E client, then VTAM must be configured to allow such LUs to be generated. A sample VTAMLST member is shown in Example 10-3 on page 257.

### Example 10-3 VTAMLST example for TN3270E LUs

---

```
* VTAMLST SAMPLE DEFINITION for TN3270 LUs
*
TELAPPL VBUILD TYPE=APPL
TCP*          APPL AUTH=NVSPACE,EAS=1,PARSESS=NO,
              MODETAB=ISTINCLM,SESSLIM=YES
LUT*          APPL AUTH=NVSPACE,EAS=1,PARSESS=NO,
              MODETAB=ISTINCLM,SESSLIM=YES
PRT*          APPL AUTH=NVSPACE,EAS=1,PARSESS=NO,
              MODETAB=ISTINCLM,SESSLIM=YES
```

---

The definitions in this member are for an APPL major node. The TN3270E server uses these application LUs to function like terminal LUs.

The LU names coded in VTAMLST must match any LU names generated via the mapping statements in the BEGINVTAM statement block.

## 10.5 Summary

The TN3270E protocol represents the evolution of SNA as it converged into the world of TCP/IP. It is the primary method of connecting end users to mainframe computers. It consists of a character-based data stream. TN3270E connectivity is handled on the mainframe by the TN3270E server. The TN3270E server converts a TN3270E TCP/IP connection to an SNA session. The terminal LU of the SNA session is emulated so that the SNA application functions as though it were connected to a non-programmable 3270 terminal.

Key terms in this chapter		
ASSOCIATE	BEGINVTAM	mapping
printer LU	TELNETPARMS	terminal LU
TN3270E	VTAMLST	3270 data stream
3270 display terminal		





# Part 4

# Network operations and administration





# Operational procedures

**Objective:** After completing this chapter, you will be able to:

- ▶ Provide an overview of the types of operational tasks that a z/OS network administrator would perform.
- ▶ Use z/OS network component commands and displays that are commonly used to monitor and control the network.
- ▶ Understand the importance and the breadth of documentation in the role of a network administrator.

## 11.1 Operational tasks

The role of a z/OS network administrator can span a wide area. In some organizations you might be a generalist, looking after all z/OS networking components, printer subsystems, and even some of the hardware. However, it is more likely that you will specialize in one or two particular areas, such as VTAM and TCP/IP.

**Note:** This chapter refers to the “network administrator” as the person who might design, control, monitor, and manage the use of the z/OS networking software and the associated network.

In some organizations, this role might be subdivided. For example, a system operator might be responsible for monitoring and controlling simple aspects of the network only, while the network administrator handles design and management. Regardless, in this chapter network administrator encompasses effectively all aspects relating to the continuous running of a network in a z/OS context.

This chapter remains at a high level, and attempts to give you a general understanding of the role.

Within z/OS networking components on the mainframe, some common tasks you are expected to fulfill include:

- ▶ Correct z/OS network-related faults.
- ▶ Change and configure network components.
- ▶ Monitor and control network components.
- ▶ Provide performance and usage network statistics.
- ▶ Work with other groups on projects, tasks and problems. These groups might include operations, the WAN network team, z/OS systems programmers, change control, business users, and testers.

“Monitoring” means watching and observing, normally to see something change. This does not imply that you sit there all day watching a screen. There are network management tools installed on z/OS to capture alert and monitor messages and events. There are also network operations staff who normally are the first level filters for problems.

**Note:** In a mainframe environment, system automation is relied upon heavily. Operational tasks are changed, and generally simplified, by a system automation package. Some commonly used system automation packages include Tivoli NetView for z/OS and System Automation for z/OS.

System automation is used not only to handle monotonous monitoring and z/OS console watching, but also for error recovery and fault tolerance.

### 11.1.1 Network startup

The network subsystems such as VTAM, TCP/IP, and related components are normally started as part of the IPL sequence under z/OS.

There is normally an automation software product that controls the startup, and this has dependency checks or parent-child relationships built into it. For example, VTAM would have to start prior to TCP/IP, and FTP (an application daemon under TCP/IP) cannot start before TCP/IP. Because FTP requires TCP/IP and TCP/IP requires VTAM, such a starting order makes sense. However, it is not always necessary. For example, TCP/IP waits for VTAM to start up if it detects VTAM is not available.

VTAM and TCP/IP are started tasks, with JCL procedures like many other z/OS components. You should familiarize yourself with the JCL and data sets, and members that are in use for these components.

The network administrator should work with the automation team to address any network component startup sequences and dependency requirements you have. Many of the network component dependencies and relationships could already be in place.

**Note:** The z/OS network administrator should develop the initial network component runbook for operations. If there are any particular checks or commands that need issuing as part of a network startup or takedown, these should be included.

#### Example dependencies

Some of the dependencies that are of concern during system startup are:

- ▶ The network cannot start before the z/OS operating system and JES2 are up and running.
- ▶ Network devices need to be varied online by the operating system.

- ▶ The network must start up *before* any network applications. There is no point in starting these before the underlying network is available.
- ▶ Enterprise Extender (EE) cannot be enabled until both VTAM and TCP/IP are up.
- ▶ OMPROUTE is started *after* TCP/IP is available.
- ▶ In a sysplex, ownership of dynamic VIPA addresses is dynamically determined, but there might be a moment during IPL when the dynamic VIPA is associated with a backup host. The same applies to a sysplex distributor environment.

## VTAM startup

Example 11-1 on page 264 contains a sample of typical startup messages for VTAM. Note that on the START command (abbreviated S, in this sample) includes the desired VTAM list data set. The systems programmer might be using z/OS symbolics to identify the LPAR.

For example, &VTAMSYMB is not actually the name of the VTAM list data set. Instead, it is a system symbol variable. In a VTAM context, each LPAR has a unique set of definitions, and startup requirements. Using symbolics can allow a different variable substitution to occur on each LPAR, but the command can be the same. For example, on and LPAR numbered 5, &VTAMSYB might get translated to:

```
S NET,,,(LIST=VTAMLST5)
```

**Note:** System symbols are defined in the z/OS parmlib member SYS1.PARMLIB(IEASYMxx). Each LPAR (z/OS image) can utilize system symbols to provide simple variable substitutions in JCL, system commands, and even within TCP/IP configuration files.

The series of IST093I messages occur when VTAM reads the ATCSTRxx start list member, and any other specified start option members, to build the VTAM environment. Message IST093I is also issued as VTAM reads the configuration list member ATCCONxx, which contains a list of VTAM resources defined by the z/OS network administrator. The list of IST093I messages are comprised of resource names that should eventually become familiar to the network administrator.

*Example 11-1 Some of the network startup messages for VTAM*

---

```
S NET,,,(LIST=&VTAMSYMB)
08.32.00 STC07376 ---- WEDNESDAY, 11 MAY 2005 ----
08.32.00 STC07376 IEF695I START NET WITH JOBNAME NET
08.32.00 STC07376 $HASP373 NET STARTED
```

```
08.32.00 STC07376 IEF403I NET - STARTED - TIME=08.32.00 -
08.32.08 STC07376 IST093I ISTCDRDY ACTIVE
08.32.11 STC07376 IST093I COSAPPN ACTIVE
08.32.11 STC07376 IST093I ISTRTPMN ACTIVE
08.32.11 STC07376 IST093I ISTTRL ACTIVE
08.32.11 STC07376 IST093I CULN830 ACTIVE
08.32.13 STC07376 IST093I APOSAXX ACTIVE
08.32.13 STC07376 IST093I APNJEXX ACTIVE
08.32.13 STC07376 IST093I APEJE2XX ACTIVE
08.32.13 STC07376 IST093I APCASXX ACTIVE
08.32.13 STC07376 IST093I APTCPXX ACTIVE
08.32.15 STC07376 IST093I OSA2380 ACTIVE
08.32.15 STC07376 IST093I OSA2800 ACTIVE
08.32.16 STC07376 IST020I VTAM INITIALIZATION COMPLETE FOR CSV1R6
08.32.16 STC07376 IST1349I COMPONENT ID IS 5695-11701-160
08.32.16 STC07376 IST1348I VTAM STARTED AS NETWORK NODE
08.32.17 STC07376 IST093I ISTLSXCF ACTIVE
```

---

One of the most important messages within Example 11-1 is message ITS020I. This message is an indication that VTAM is now ready to receive and process VTAM network operator commands.

### **TCP/IP startup**

The TCP/IP procedure (or procedures) might also be started through automation. A sample is shown in Example 11-2 on page 266. Unlike VTAM, TCP/IP does not take any parameters with respect to its configuration (other than tracing options).

Instead, the TCP/IP task locates its parameters from the PROFILE DD statement within the startup JCL itself. The profile is read, as outlined by the EZZ0309I and EZZ0316I message pair. Afterwards, significant configuration attributes found within the TCP/IP profile are identified with a series of messages.

The EZZ4202I message is of some interest; TCP/IP not only depends upon VTAM, but it also requires that z/OS UNIX System Services be active and functioning.

*Example 11-2 Some of the TCP/IP startup messages*

---

```
S TCPIP
08.32.34 STC07499 ---- WEDNESDAY, 11 MAY 2005 ----
08.32.34 STC07499 IEF695I START TCPIP WITH JOBNAME TCPIP
08.32.34 STC07499 $HASP373 TCPIP STARTED
08.32.56 STC07499 EZZ0300I OPENED PROFILE FILE DD:PROFILE
08.32.56 STC07499 EZZ0309I PROFILE PROCESSING BEGINNING FOR DD:PROFILE
08.32.58 STC07499 EZZ0316I PROFILE PROCESSING COMPLETE FOR FILE DD:PROFILE
08.32.58 STC07499 EZZ0641I IP FORWARDING NOFWMULTIPATH SUPPORT IS ENABLED
08.32.58 STC07499 EZZ0335I ICMP WILL IGNORE REDIRECTS
08.32.58 STC07499 EZZ0350I SYSPLEX ROUTING SUPPORT IS ENABLED
08.32.58 STC07499 EZZ0352I VARIABLE SUBNETTING SUPPORT IS ENABLED FOR OROUTED
08.32.58 STC07499 EZZ0624I DYNAMIC XCF DEFINITIONS ARE ENABLED
08.32.58 STC07499 EZZ0338I TCP PORTS 1 THRU 1023 ARE RESERVED
08.32.58 STC07499 EZZ0338I UDP PORTS 1 THRU 1023 ARE RESERVED
08.32.59 STC07499 EZZ4202I Z/OS UNIX - TCP/IP CONNECTION ESTABLISHED FOR TCPIP
08.32.59 STC07499 EZZ4313I INITIALIZATION COMPLETE FOR DEVICE IUTSAMEH
08.32.59 STC07499 EZZ4313I INITIALIZATION COMPLETE FOR DEVICE OSA2380
08.33.01 STC07499 EZZ4313I INITIALIZATION COMPLETE FOR DEVICE IUTIQDIO
08.33.03 STC07499 EZB6473I TCP/IP STACK FUNCTIONS INITIALIZATION COMPLETE.
08.33.03 STC07499 EZAIN11I ALL TCPIP SERVICES FOR PROC TCPIP ARE AVAILABLE.
08.33.04 STC07499 EZZ0400I TELNET/VTAM (SECOND PASS) BEGINNING FOR FILE:
08.33.05 STC07499 EZZ6003I TELNET LISTENING ON PORT 23
08.33.05 STC07499 EZZ0403I TELNET/VTAM (SECOND PASS) COMPLETE FOR FILE:
08.33.05 STC0749 S OMPROUTE
08.33.05 STC0749 S FTPD
```

---

As with VTAM, TCP/IP has a message indicating that it has completed its startup and is ready to accept operator commands: EZAIN11I.

Note the messages that continue after EZAIN11I. The TN3270E server can be configured as part of the TCP/IP address space, as is the case in this example. After TCP/IP has started up, the TN3270E server configuration statements are then processed. The EZZ4313I messages are of significant interest here—a network administrator would want to make certain that all devices come active during the startup.

Finally, in this sample, TCP/IP has been configured to automatically start (AUTOLOG) the OMPROUTE and FTPD started tasks. This could also be accomplished through an external automation package.

**Tip:** How might external automation work? In Example 11-2, an automation package could be configured to look for message EZAIN11I. As soon as the automation software finds that message, it could then issue a start of OMPROUTE and FTPD, and any other software that depends upon TCP/IP.

## 11.1.2 Stopping the network

A network shutdown would normally be a process that occurs as part of a scheduled z/OS IPL or swapping of the network from one LPAR to another. A scheduled IPL occurs during a period of little activity on the host and network. Usually, this means it occurs in the evening hours of a weekend.

The reasons for requiring a scheduled IPL are becoming fewer and fewer since hot-swappable devices and components are becoming prevalent on the mainframe. Some maintenance still requires an IPL, however, and there are also organizations that do scheduled IPLs as part of a regular routine. Regardless, if an LPAR is shut down, that means the network components must also be shut down.

A network shutdown might be independent of an LPAR shutdown. For example, in a partial network shutdown, TCP/IP or VTAM might be taken down independently. In such a situation, there might be a VIPA takeover process that might occur.

It could be a dynamic VIPA that moves automatically to a running LPAR in the sysplex. It could be a static VIPA that is moved by the network administrator during the shutdown. Or, it could be an external automation package moving the static VIPA automatically. Client connections might never be aware of the changes occurring.

Because TCP/IP depends upon VTAM, a shutdown of VTAM not only ceases all SNA communications, it also halts all IP communications. TCP/IP automatically detects that VTAM is no longer available and waits for it to be restarted.

The order in which the network is shut down is the reverse of a startup. Generally, it is:

1. TCP/IP and VTAM applications are stopped. Each application has its own shutdown command.
2. TCP/IP is stopped through: P TCPIP
3. VTAM is then stopped through: Z NET,QUICK

This process is normally performed through automation. Note that when halting network VTAM and TCP/IP communications, the network administrator must ensure that access to the console is not also halted!

## 11.2 z/OS network administrator tasks

Network administrator tasks can be varied and are, one way or another, derived from the needs of the organization within which the network administrator functions. In an environment other than z/OS, a network administrator might have a role that encompasses many platforms and hardware areas.

However, z/OS network administrators tend to have a more narrow focus. One reason for this is complexity: network administration on z/OS requires a good working knowledge of z/OS itself. It also requires a good knowledge of networking hardware. When you combine this with the actual VTAM, TCP/IP, LAN, and WAN knowledge required, the z/OS network administrator is unlikely to have the opportunity to include other platforms.

So what are some of the tasks a z/OS network administrator might undertake? A sampling is described in Table 11-1.

*Table 11-1 Network administrator tasks*

<b>Task</b>	<b>Description</b>
Problem source identification	When something does not function as expected, the network administrator is one of the first persons to work towards resolution.
Network control	VTAM, TCP/IP, and their associated applications must be started, stopped, monitored, and maintained as required.
Planning	Planning includes network architecture decisions, such as what role z/OS plays in the network and how z/OS should be situated in the network.
Change control	In a z/OS network environment, all changes are part of a planned and controlled process. The z/OS network administrator would work with other network administrators.
Hardware evaluation	If a new feature is to be added to the z/OS host or to the network used by the z/OS host, then an evaluation of the impact of the feature must be assessed.
Software evaluation	In the TCP/IP world, new networking applications and updated existing ones are literally a daily phenomenon.
Installation of hardware and software	Once the decision is made, the actual software or hardware must be implemented.
Capacity planning	Network capacity requirements are always changing.

Task	Description
Paperwork	Documentation of the environment, changes, and procedures are all part of the network administrator's role.

## 11.2.1 Managing VTAM

The following are examples of some of the VTAM commands that you might use in a z/OS network administration role for controlling VTAM. Refer to *z/OS Communications Server SNA Operation*, which contains VTAM operations and commands, for detailed information about commands and examples.

Like all z/OS environments, VTAM allows the use of some abbreviations instead of the full command syntax. Some automation software products have taken this a step further and provided command executables that eliminate the need to know the full VTAM command syntax.

### Displaying VTAM resources

As previously mentioned, Enterprise Extender (EE) is being used by many customers to carry SNA traffic over IP networks. One of the most commonly used commands to check the status of an EE network is illustrated in Example 11-3.

The DISPLAY NET,EE command (DISPLAY can be abbreviated as D) gives the network administrator a quick look at the remote IP endpoint of an EE session. In this sample, the remote endpoint was 192.168.80.90. The command output provides the line and PU (physical unit) information as well as other node characteristics.

You can see some information on the connection beginning at IST2035I. NLP stands for *network layer packet*. The NLPs retransmitted count can be used to gauge the amount of lost traffic over the session.

#### *Example 11-3 DISPLAY of remote Enterprise Extender endpoint*

---

```
D NET,EE,IPADDR=(,192.168.80.90)
IST097I DISPLAY ACCEPTED
IST350I DISPLAY TYPE = EE
IST2001I ENTERPRISE EXTENDER CONNECTION INFORMATION
IST924I -----
IST1680I LOCAL IP ADDRESS 192.168.80.9
IST1910I LOCAL HOSTNAME VIPA16.SSCP1A.TCP.RALEIGH.IBM.COM
IST1680I REMOTE IP ADDRESS 192.168.80.90
IST1909I REMOTE HOSTNAME VIPA16.SSCP2A.TCP.RALEIGH.IBM.COM
```

```

IST2022I EE CONNECTION ACTIVATED ON 09/12/05 AT 09:27:15
IST2023I CONNECTED TO LINE LNEE1000
IST2024I CONNECTED TO SWITCHED PU SWEE2A1
IST2025I LDLC SIGNALS RETRANSMITTED AT LEAST ONE TIME = 0
IST2026I LDLC SIGNALS RETRANSMITTED SRQRETRY TIMES = 0
IST2009I RTP PIPES = 7 LU-LU SESSIONS = 6
IST2027I DWINOP = NO REDIAL = *NA* REDDELAY = *NA*
IST2028I KEEPACT = NO
IST2029I MTU SIZE = 1232
IST924I -----
IST2035I TOTALS FOR ALL PORT PRIORITIES
IST2036I NLPS SENT = 95182 ( 095K )
IST2037I BYTES SENT = 10787917 ( 010M )
IST2038I NLPS RETRANSMITTED = 0 ( 000K )
IST2039I BYTES RETRANSMITTED = 0 ( 000K )
IST2040I NLPS RECEIVED = 95227 ( 095K )
IST2041I BYTES RECEIVED = 9756692 ( 009M )
IST2042I 1 OF 1 EE CONNECTIONS DISPLAYED
IST314I END

```

---

A network administrator might want information on a reported problem with an APPN RTP (rapid transport protocol) link. The administrator would use the DISPLAY NET,RTPS command.

In Example 11-4, the RTP PU cnr00004 is being tested. An RTP PU represents the endpoint of an HPR route. The result of this command is a display, in message IST1792I, of the total time taking to traverse the route. Routing delays in SNA can be of significant impact for time sensitive applications. The advantage of this test command is that the network administrator might be able to determine quickly whether the reported problem is really with the link, or whether the delay is being introduced by an application.

*Example 11-4 DISPLAY of the HPR route test*

---

```

D NET,RTPS,ID=cnr00004,TEST=yes
IST097I DISPLAY ACCEPTED
IST350I DISPLAY TYPE = RTPS
IST1695I PU NAME      CP NAME      COSNAME SWITCH CONGEST STALL SESS
IST1960I CNR00004  NETA.SSCP2A  #INTER  NO      NO      NO      1
IST1786I HPR ROUTE TEST INITIATED FOR RTP PU
IST1454I 1 RTP(S) DISPLAYED
IST314I END
IST1787I HPR ROUTE TEST RESULTS FOR RTP PU CNR00004
IST1788I NODE CP NAME  TG NUMBER  PARTNER CP NAME  INTERNODAL TIME
IST1789I                                     (MILLISECONDS)
IST1790I NETA.SSCP1A   21         NETA.SSCP2A     1
IST1792I TOTAL RTP TRAVERSAL TIME 1 MILLISECONDS

```

## Changing VTAM resources

The primary command used for controlling VTAM is the VARY command.

**Note:** Ensure you understand the Vary and Modify commands well before using them. Your security profile as a z/OS network administrator provides you with the ability to fully control the network and its resources. Always verify that the resource you are changing or modifying is the correct one—and understand the impact.

### ***VARY command***

The VARY command affects the status of the physical resource and session. Usually, a vary command results in a complete state change from active to inactive or the opposite. The vary command is abbreviated as the single character V.

The following command activates a VTAM resource that might be in an inactive state or a new VTAM resource you have just created:

```
V NET,ACT,ID=resource
```

#### *Example 11-5 The VARY ACT command*

---

```
V NET,ACT,ID=Z6TMD005  
IST097I VARY ACCEPTED  
IST093I Z6TMD005 ACTIVE
```

---

The following command inactivates a VTAM resource that might be in an active state, have a problem, or needs to be made inactive in order to modify it, and then reactivate it:

```
V NET,INACT,ID=resource
```

#### *Example 11-6 The VARY INACT command*

---

```
V NET,INACT,ID=Z6TMD005  
IST097I VARY ACCEPTED  
IST105I Z6TMD005 NODE NOW INACTIVE
```

---

### ***The MODIFY command***

The MODIFY command enables you to change VTAM options, tables, storage and traces. Usually, a modification command results in a change in the

characteristics of a resource, but does not change it from a state of active to inactive or the opposite.

The modify command is abbreviated as the single character F:

```
F NET,VTAMOPTS,startoption=option
```

This command allows you to modify VTAM start options; with some options there are restrictions and dependencies. If your VTAM was defined as an interchange node (ICN) containing both subarea and APPN support, you might want to change the VTAM search order (SORDER) option.

If the current setting was set to SUBAREA, you might want to change it to APPNFRST. You would do this with the following command:

```
F NET,VTAMOPTS,SORDER=APPNFRST
```

The following command enables you to turn on a VTAM buffer for a particular resource:

```
F NET,TRACE,id=resource,TYPE=BUF
```

The following command stops the VTAM buffer trace:

```
F NET,NOTRACE,ID=resource,TYPE=BUF
```

Example 11-7 shows the use of these commands.

*Example 11-7 VTAM buffer trace started, then stopped on a VTAM LU*

---

```
F NET,TRACE,ID=Z6TS0001,TYPE=BUF
IST097I MODIFY ACCEPTED
IST1515I BUFFER TRACE ACTIVE
IST513I TRACE INITIATED FOR NODE Z6TS0001
F NET,NOTRACE,ID=Z6TS0001,TYPE=BUF
IST097I MODIFY ACCEPTED
IST512I TRACE TERMINATED FOR NODE = Z6TS0001
```

---

### ***The HALT command***

This command is used to stop VTAM—which is not something you want to test on your production LPAR! A HALT command without any parameters performs a nondisruptive end to VTAM. The HALT command is abbreviated as the single character Z.

The following command halts VTAM more quickly than a normal HALT command. The command causes sessions to be terminated and VTAM to shut down.

```
Z NET,QUICK
```

The following command causes VTAM to abend and is quite a nasty way to bring it down. You would normally try the Z NET,QUICK command first.

```
Z NET,CANCEL
```

## 11.2.2 Managing TCP/IP

TCP/IP, like VTAM, has a number of commands available to monitor and change the environment.

TCP/IP provides two methods of issuing most commands, either through the z/OS console or from within a TSO or z/OS UNIX environment. However, the more powerful commands for controlling the TCP/IP environment are issued from the z/OS system console. This section focuses on console commands only. The TCP/IP commands are documented in *z/OS IP System Administrator's Commands*.

### Displaying TCP/IP

There are many commands for displaying and testing a TCP/IP network. Commands such as PING and TRACERTE are not covered here. They are effectively standard across all platforms and commonly used.

The DISPLAY TCPIP,NETSTAT command is often used to monitor TCP connections. On a busy host, the command would need to be narrowed in focus to only display connections for a specific port. For example, a display of all TN3270 connections (port 23) can be found in Example 11-8.

The complete command issued was:

```
D TCPIP,tcpprocname,NETSTAT,ALLCON,PORT=23
```

In this display, a network administrator can determine at a glance the number of connections and the state of each connection. If a problem is noted, the IP address of the remote host (FOREIGN SOCKET) is readily available.

*Example 11-8 DISPLAY NETSTAT of port 23*

---

```
EZZ2500I NETSTAT CS V1R6 TCPIP
USER ID  CONN      LOCAL SOCKET          FOREIGN SOCKET        STATE
TCPIP    00000017  0.0.0.0..23          0.0.0.0..0           LISTEN
TCPIP    0000D4DE  192.168.246.24..23   192.168.6.251..2119  ESTBLSH
TCPIP    00244B11  192.168.246.24..23   192.168.7.99..1499   CLOSWT
3 OF 3 RECORDS DISPLAYED
```

---

On a regular basis, some automation programs check device or route status using the DISPLAY NETSTAT command. A network administrator might do the same manually, or might do so in response to a reported connectivity problem.

To check the status of all TCP/IP devices, the following command might be issued:

```
D TCPIP,tcpprocname,NETSTAT,DEV
```

Example 11-9 shows the output from the command. Note the LNKSTATUS field, which describes whether the device is fully functional (READY) or not. In addition, various statistics and counts can be found.

*Example 11-9 DISPLAY NETSTAT DEV command output*

---

```
DEVNAME: OSA100          DEVTYPE: MPCIPA
DEVSTATUS: READY        CFGROUTER: NON  ACTROUTER: NON
LNKNAME: OSATR100      LNKTYPE: IPAQTR   LNKSTATUS: READY
NETNUM: N/A  QUESIZE: N/A  SPEED: 0000000100
MACADDRORDER: NON-CANONICAL  SRBRIDGINGCAPABILITY: YES
IPBROADCASTCAPABILITY: NO    ARPROADCASTTYPE: ALL RINGS
ARPOFFLOAD: YES              ARPOFFLOADINFO: YES
ACTMTU: 17914
READSTORAGE: GLOBAL (4096K)  INBPERF: BALANCED
BSD ROUTING PARAMETERS:
MTU SIZE: 00000            METRIC: 00
DESTADDR: 0.0.0.0          SUBNETMASK: 255.255.255.128
MULTICAST SPECIFIC:
MULTICAST CAPABILITY: YES
GROUP          REFCNT
-----
224.0.0.1      0000000001
LINK STATISTICS:
BYTESIN                    = 7721257
INBOUND PACKETS            = 44830
INBOUND PACKETS IN ERROR   = 85896
INBOUND PACKETS DISCARDED  = 0
INBOUND PACKETS WITH NO PROTOCOL = 0
BYTESOUT                   = 252
OUTBOUND PACKETS           = 3
OUTBOUND PACKETS IN ERROR  = 0
OUTBOUND PACKETS DISCARDED = 0
1 OF 1 RECORDS DISPLAYED
```

---

If a connectivity problem is apparent and the device appears to be in good working order, the next step would be to ensure that there were no routing problems with respect to (or from the perspective of) the z/OS host. Obviously, a routing problem could exist elsewhere in the network.

The command to display routes is:

```
D TCPIP,tcpprocname,NETSTAT,ROUTE
```

Example 11-10 on page 275 shows sample output from a NETSTAT ROUTE command. A network administrator would need to know which network is experiencing connectivity problems. The network administrator would then scan this table for the appropriate route taken to reach the network.

The flags field tells the network administrator information about the route. For example, in this display, the flags shown are:

- U** The route is Up.
- G** The route is a gateway (default) route.
- S** The route was created from a static definition (not learned dynamically via RIP or OSPF, for example).
- H** The route is a host route, with a network mask of 32 bits (that is, only one destination host is defined by this route).

Other route flags exist and can be found documented under the NETSTAT command in *z/OS IP System Administrator's Commands*.

*Example 11-10 DISPLAY NETSTAT ROUTE command*

---

```
EZZ2500I NETSTAT CS V1R6 TCPIP 278
DESTINATION      GATEWAY          FLAGS      REFCNT  INTERFACE
DEFAULT          192.168.246.1    UGS        000013  ETH1
DEFAULT          192.168.247.129 GS          000000  OSATR104
9.21.111.210/32  0.0.0.0          H          000000  SNA1
9.23.246.0/24    0.0.0.0          US         000000  ETH1
9.23.246.0/24    0.0.0.0          S          000000  ETH2
9.23.246.16/32   0.0.0.0          UH         000000  VLINK1
9.23.246.23/32   0.0.0.0          H          000000  ETH2
9.23.246.24/32   0.0.0.0          UH         000000  ETH1
9.23.247.128/25  0.0.0.0          US         000000  OSATR100
9.23.247.128/25  0.0.0.0          S          000000  OSATR104
9.23.247.130/32  0.0.0.0          H          000000  OSATR104
9.23.247.135/32  0.0.0.0          UH         000000  OSATR100
127.0.0.1/32     0.0.0.0          UH         000020  LOOPBACK
13 OF 13 RECORDS DISPLAYED
```

---

## Controlling TCP/IP

You can alter the TCP/IP configuration with the VARY TCPIP,,OBEYFILE command. The OBEYFILE command is a very powerful command, since it can change any aspect of the TCP/IP configuration. This is because the OBEYFILE command runs a process similar to the process that runs during TCP/IP startup. An OBEYFILE causes TCP/IP to read a configuration file in the same fashion as the TCP/IP PROFILE data set is read at startup.

A sample of an OBEYFILE command is as follows:

```
VARY TCPIP, tcpprocname, OBEYFILE, DSN=your.obey.file
```

**Note:** Unlike VTAM, of which only one copy can ever execute within a single LPAR, up to 8 TCP/IP stacks can be running concurrently within an LPAR. To identify which stack should be the target of a command, *tcpprocname* should be included in any TCP/IP command.

If *tcpprocname* is not coded (that is, there is nothing between the two commas), then the operand defaults to TCPIP.

In this example, *your.obey.file* might contain a new set of DEVICE and LINK statements, a HOME statement, or both.

Sometimes TCP/IP connections need to be cleared by issuing a VARY TCPIP,,DROP command. Before issuing the command, a network administrator would have discovered a connection that needed to end. Either the connection is an undesired connection entirely, or else it is in a state needing external intervention. Such a connection would be discovered using the DISPLAY NETSTAT command illustrated in Example 11-8 on page 273.

Note in this example that the last connection is in a state of CLOSWT. A CLOSWT connection is often an indication of a problem with a TCP/IP application. The application might have left this connection in this unusable state indefinitely. The DROP command can be used to cleanup (remove) such a connection. The command to remove the CLOSWT connection above would need to use the connection number to identify it on a DROP command, as follows:

```
VARY TCPIP, tcpprocname, DROP, CONN=244B11
```

This command would clean up the connection as far as the z/OS TCP/IP is concerned.

The VARY TCPIP,,STOP and VARY TCPIP,,START commands are used for stopping or starting TCP/IP devices. These commands are often the first set of commands issued when a network administrator discovers a device failure. For example, if a device problem was detected and the device was known to be inoperative, a VARY STOP followed by a VARY START would be attempted.

**Note:** OSA devices can be configured with an AUTORESTART parameter to cause TCP/IP to attempt to restart automatically a device under most circumstances.

A sample command would be:

```
VARY TCPIP, tcpprocname, START, OSA2380
```

This command causes TCP/IP to attempt to restart device OSA2380.

### 11.2.3 Controlling TCP/IP applications

There are individual commands to control the more significant (that is, well known and well used) TCP/IP applications. There are MODIFY commands to control the IKE server, OMPROUTE, the policy agent SNMP and others. In this section, the FTP server and TN3270E server are discussed.

#### FTP server

The FTP server supports the ability dynamically to activate and to deactivate several different debugging options through the MODIFY command.

```
MODIFY ftpdjobname,DEBUG=(parm)
```

The DEBUG parameter can take several different operands such as:

<b>CMD</b>	Trace of FTP subcommand activity
<b>SEC</b>	Trace of FTP security (TLS) functions
<b>SOC</b>	Trace of FTP socket activity
<b>NONE</b>	Turn off FTP server tracing

There are other operands available, as well as the ability to trace a specific user ID. For example, the command to activate tracing of FTP subcommand and security processing for the user ID MATT is:

```
F FTPD1,DEBUG=(CMD,SEC,USERID(MATT))
```

Interpretation of an FTP trace requires in-depth knowledge of both z/OS and FTP. Often, FTP traces are taken at the advice of IBM service personnel, who also handle the interpretation of the output.

#### TN3270E server

The command suite for controlling the TN3270E server is large for two reasons. First, the TN3270E server is the foremost method of communicating with the mainframe. Second, the TN3270E server straddles SNA and IP, which means it must operate with consideration for both environments.

#### Displaying the TN3270E server

The DISPLAY TCPIP,*telenetprocname*,TELNET command supports a large suite of operands that affords the network administrator either a wide or narrow view of the TN3270E server. Operands can produce summaries or detailed displays about individual aspects of the server.

For example, if a network administrator wanted a complete view of the configuration of an active TN3270E server, the following command could be used:

```
D TCPIP, TELNETS, TELNET, PROFILE, DETAIL
```

The resulting output from this command has been edited for brevity and can be seen in Example 11-11 on page 278. Of interest are such fields as the following:

<b>PERSISTENCE</b>	This section includes the LUSESSIONPEND parameter, which controls whether the user is returned to a logon-type panel after logging off of an application.
<b>DIAGNOSTICS</b>	This section shows what tracing is active.
<b>SECURITY</b>	This section shows the type of connections allowed. A BASIC connection contains no TLS security.
<b>TIMERS</b>	The inactivity timers are important to the TN3270E server in event of network outages or clients that fail to indicate the connection has ended.

The total number of records displayed is indicated at the bottom of this display command. This simple command resulted in 85 records. This command provides considerably more detail about the active TN3270E server's profile than what has been shown.

*Example 11-11 DISPLAY TELNET PROFILE command (edited)*

---

```
EZZ6080I TELNET PROFILE DISPLAY
....
PERSISTENCE
  NOLUSESSIONPEND
  NOMSG07
  NO TKOSPECLU
  NO TKOGENLU
  NO QUEUESESSION
  NODROPASSOCPRINTER
  KEEPLU                0 (OFF)
....
DIAGNOSTICS
  DEBUG                EXCEPTION
  DEBUG ROUTING        CONSOLE
  NOFULLDATATRACE
SECURITY
  PORT                23
  CONNTYPE            BASIC
  KEYRING             **N/A**
....
TIMERS
  INACTIVE            0 (OFF)
```

```
KEEPINACTIVE          0 (OFF)
PRTINACTIVE           0 (OFF)
```

---

The other commonly use DISPLAY command is the following:

```
D TCPIP, telnetprocname, TELNET, CONN
```

Example 11-12 shows the output from this command. Using this command is a quick way for a network administrator to determine all active connections to the TN3270E server. It is similar to the DISPLAY NETSTAT command with a filter on port 23, except that basic TN3270E information is included.

The field TSP PTR stands for terminal, session, protocol pointer. The terminal column describes whether the connect is for a terminal (T) or printer (P) (all connections are for terminals, in this example).

The session column indicates whether the session is active (A), pending (P), or negotiating (N) (not shown in this example). The last column, protocol, is normally for TN3270E, but sometimes a 3 could show up here, indicating a TN3270 connection is in use (there are other possible protocols, but they are extremely rare).

*Example 11-12 DISPLAY TELNET CONNECTION command*

---

```
EZZ6064I TELNET CONNECTION DISPLAY
          EN
CONN     TY IPADDR..PORT          LUNAME  APPLID  TSP
-----  -  -
000188B7 192.168.105.20..1568  OVHTCP27 OVHSAMON TAE SNX32705
000171B3 192.168.185.29..3554   OVHTCP21 OVHTS017 TAE SNX32702
00015AF1 192.168.185.90..3183   OVHTCP14 OVHTS027 TPE SNX32705
000157CA 192.168.185.79..3008   OVHTCP12 OVHTS038 TAE SNX32702
-----  -  -
----- PORT:    23  ACTIVE          PROF: CURR CONNS:    4
-----  -  -
6 OF 6 RECORDS DISPLAYED
```

---

## Controlling the TN3270E server

The TN3270E server also supports extensive control commands:

```
VARY TCPIP, telnetprocname, ACT, luname
```

Commands for activating and inactivating LUs are available. However, they are not the same as a VTAM activation of a resource. A command to inactivate a TN3270 LU simply makes an LU unavailable to the TN3270E server; it has no effect on the resource status. Such a command might be used by the network administrator in the event of an LU causing connectivity problems. To inactivate LU OVHTCP12, the following command could be used:

V TCP/IP, *telnetprocname*, INACT, OVHTCP12

At startup of the TN3270E server, all LUs are considered active (available) by default.

There are also commands to stop and restart TN3270E services without having to actually stop or restart the TN3270E server.

## 11.3 Documenting the network environment

Documentation is something that people like to read—but typically do not like to create! Good documentation is worth its weight in gold. It is a great tool for learning about your network environment—and it helps to reduce the time it takes to resolve a problem.

The IBM VTAM and TCP/IP network product manuals are very detailed. Such proportionate detail might be necessary in regard to the processes, diagrams and setup information relevant to your organization. The information found in this type of documentation might include:

- ▶ Network component overview diagram

Types of devices, protocols in use, mainframe network interfaces, LPAR names, TCP/IP addresses, network and subnetwork addresses, VTAM SSCP names, data center switches, and routers the mainframe is connected to. This information might not be shown all on one diagram, but somewhere it must all be documented and available.

- ▶ Network component description

Describe the components in the diagram (such as VTAM, TCP/IP, routing, and interfaces) and explain how they are defined within your organization. Include IP network and subnetwork (network ID) information.

- ▶ Application descriptions

Document application names and describe how they connect to the network.

- ▶ External connections

Include details about connections to other organizations, as well as a brief description of what the connections are used for. Also include WAN service provider details, the protocol used, and the equipment used by the WAN service.

Remember to cover the existence of any virtual private network (VPN) capabilities to external sites.

- ▶ Network naming conventions

Document the data set names of source libraries, and where to find started task procedures, VTAM names of devices, applications and resources. TCP/IP naming conventions might include information about host names.

- ▶ Network processes

Document the processes relevant to the networking role, common tasks you might have to perform such as problem diagnosis, change control, call out procedures, and describe where to find additional information.

- ▶ Network-related products, tools, exits, and automation

Document the network controlling and monitoring products, as well as exits that might have been implemented and why. Also cover automation dependencies, and explain how to start or stop components manually.

- ▶ Change log

Keep a log of all changes, describing when, what, and why a change was undertaken.

- ▶ Problem log

Some organizations want you to record any network issues you might have, and to document their resolution.

- ▶ Contact details

Document the people and group contacts that you work with, either in changes or problems, internally and externally.

- ▶ Security policy

A security policy should never be far from a network administrator's mind. The network generally represents the most vulnerable aspect of a host. Any changes and all processes must be in accordance with an organization's security policies. Consequently, any or all of the documentation described here might include information on security classification or usage guidelines.

There are many other categories that could be included here. It sometimes seems as though documentation is more work than is worthwhile.

However, documentation is an integral part of a high availability network and host environment. Some say the document should be a living and breathing thing - the network rarely stands still!

## 11.4 Summary

Key terms in this chapter		
DISPLAY (D) commands	FTP server	HALT (Z) command
LU name	MODIFY (F) command	NETSTAT command
OBEYFILE command	TN3270E	VARY (V) command
VTAM		



# Network security

**Objective:** After completing this chapter, you will be able to:

- ▶ Explain what security means in a data processing context.
- ▶ Give reasons why security is of importance.
- ▶ Demonstrate the actions that can be taken to improve network security with TCP/IP on z/OS.

## 12.1 Security

Security is all about how to reduce the impact of either intentional or unintentional damage. Damage is something most of us try to avoid. If you're a car lover, there are actions you take to prevent your car from being stolen, like attaching a wheel lock, setting the car alarm, and locking doors. To prevent unintentional damage to your car, you might park it in your garage, away from traffic.

In addition, security includes the actions taken once damage has, or is believed to have, occurred. If your car is stolen, you call the police. If your car door gets "dinged", you have a body shop fix it.

Planning for security includes asking questions like "What if this happens?" or "How do I prevent that from happening?" Security also includes answering the question "Do I really want to go through all this work in order to be more secure?" Security does not always cover all possibilities, and this can be a conscious decision. It is possible to break an encrypted session's code and decrypt the data illicitly. However, that is highly improbable. Is highly improbable good enough? It would be nice to answer yes, but the truth is: it depends.

### 12.1.1 The context of security

Prior to dealing with the implementation of security, a significant amount of planning is required. Generally, a large organization will create (and continually update) a security policy document. A security policy document is an executive-level document that includes such information as:

- ▶ The classification of security levels for the company's data
- ▶ How different data is to be classified within these levels
- ▶ Processes and procedures relating to all security aspects of an organization (physical, legal, administrative, and more)

In tandem with, or as a result of, a security policy, hosts and networks will have a classification. For example, most z/OS hosts will be in secure physical locations with strict authorization requirements for employee access. Some data on such a z/OS host would most likely be considered highly secure and could be placed without any encryption on disk. If the data was to be transferred between two z/OS hosts within that secure physical location, the data transfer would most likely be classified as highly secure.

If that data were to move outside onto the organization's network, the data would now likely be considered in a lower security context. Then, if the data were placed onto a workstation at a user's home office, the classification of the data

would still have not changed. However, most would agree that its context was now a non-secure, or certainly less secure, one.

So, if a question is asked “Should I be using such-and-such a security measure with my data?” the answer is “It depends.” With a security policy in place, an organization is equipped to determine those dependencies and make a recommendation.

## 12.2 Elements of security

In the car example cited at the start of this chapter, there were several measures taken by the owner to improve security. A steering wheel lock is intended to render the car unusable. Analogous to that is the *encryption* of data, which renders the data unusable. Parking the car in a garage reduces the likelihood of physical damage to the vehicle, which is analogous to *physical site security*. Activating the car alarm constitutes a security monitoring system, usually referred to as *intrusion detection services* within a network context. Locking the vehicle is intended to enforce *authentication*: if you don’t have the right key for the car, you aren’t authorized to access it. Finally, in the event of a security breach, there must be an action plan in place: for a stolen car, call the police; for stolen data, call your manager and the database guy.

Table 12-1 lists some elements of security in an information technology context.

Table 12-1 Elements of security

Elements of security	Concerns
Identification and authorization	Who are you and can you prove your identity?
Authorization	Who is allowed to access this network, data, or system?
Monitoring and auditing	Has the system, network, or data been accessed, and is this access potentially dangerous?
Data confidentiality and integrity	Can this data be viewed or altered without permission?

In the following section we discuss these security elements as they relate to TCP/IP on z/OS. There are other elements in security, such as administration, legal considerations, recoverability, physical (site) security, and awareness. However, these topics are not covered here. And, as mentioned before, the choice of implementation depends on an organization’s requirements.

## 12.3 TCP/IP security

The security features relating to TCP/IP on z/OS are extensive. When combined with all the security capabilities of the System Authorization Facility (SAF) interface, the amount of control that can be exercised is phenomenal.

**Reminder:** The SAF interface is a standardized function call available to all applications running on z/OS. The interface call is used to provide quick and controlled authorization, authentication, and logging services. The SAF call is forwarded to an external security manager such as the Resource Access Control Facility (RACF).

Note that the term “external” refers to the fact that the security management is an independent entity outside of the currently executing application’s environment. The external security manager manages a secure database that is used to verify the security information as it relates the user ID active when the SAF request is made.

It would not be hard to configure a system that was a paragon of security. However, such a system could also become unmanageable. There is a cost in terms of usability and manageability with every security feature activated. So, the fact that a security feature exists is great, but it certainly does not mean it needs to be put into effect. Remember, “it all depends.”

Keep in mind that the security options discussed here do not represent a complete list. Some features that are seemingly unrelated to security may inadvertently enhance it. For example, z/OS clustering represents an availability improvement; refer to Chapter 6, “TCP/IP in a sysplex” on page 103 for more information about this topic.

Availability improvements are very much a form of improved security. If a successful attack is made against an individual host in a cluster of computers, the presumption is that one of the other hosts in the cluster can make up for the missing host.

Some security features have somewhat overlapping effects. IP filtering and network access can both be used to prevent certain packets from reaching their destination. SSL and IPSec both result in improved confidentiality and integrity of data.

## 12.3.1 Industry standard security features

The term *industry standard* is used to describe security features of z/OS that are widely implemented and either formally or informally standardized. One item discussed, Transport Layer Security (TLS), will be treated in some detail because it is probably the most widely used security protocol on the Internet today. And in order to implement it on z/OS, it is a concept that should be fully understood.

### Application layer security

Although application layer security is not standardized, there is one application layer form of security that approaches an industry standard: user ID and password authentication.

On z/OS, the authorizations granted to an end user are all associated with the active user ID. When a user logs onto telnetd for example, a SAF call is made to verify that the password supplied matches that of the user ID. Once verified, this user ID becomes associated with a security environment that lasts the duration of the session. SAF products such as RACF allow the creation of specific password rules, forcing them to be of a minimum length, to be renewed regularly, to be not repeated, and to contain a variety of character types.

Although it is often taken for granted, the user ID and password method of authorization is still arguably the best.

### Transport Layer Security (TLS)

Secure Sockets Layer (SSL) is a protocol standard developed by the Netscape Communications Corporation that uses encryption to provide confidentiality and authentication between two TCP/IP applications. As SSL gained in popularity, the IETF formally standardized SSL, made a few improvements and changed the name to Transport Layer Security (TLS). TLS is defined in Request for Comments (RFC) 2246.

For simplicity, the acronym TLS is used to denote TLS and SSL in this chapter.

A TLS connection begins with a *handshake*. As the name suggests, the handshake entails the initial setup of a TLS connection. During the TLS handshake, an exchange of information occurs that includes the following (this has been simplified):

- ▶ Authentication of the server
- ▶ A decision about how the data is to be encrypted
- ▶ Optionally, the authentication of the client

On z/OS, the FTP server, the FTP client and the TN3270 server all support TLS. Configuration of TLS is similar in all three environments, and the principles

involved are common throughout as well. RFC 2246 describes TLS in general—but FTP has the added complexity of a control and data connection.

To clarify how FTP handles TLS, RFC 4217 has been published. An easy way to introduce the principles is by beginning with statements in the FTP.DATA file required to support a TLS connection; see Example 12-1.

*Example 12-1 FTP.DATA configuration statements*

---

```
CIPHERSUITE SSL_DES_SHA           ;for testing, keep these cipher
CIPHERSUITE SSL_RC2_MD5_EX        ;cipher statements
CIPHERSUITE SSL_RC4_MD5_EX        ;in this order
CIPHERSUITE SSL_3DES_SHA          ;export restricted?
SECURE_FTP REQUIRED
SECURE_LOGIN NO_CLIENT_AUTH
SECURE_CTRLCONN PRIVATE
SECURE_DATACONN PRIVATE
KEYRING FTPPring
```

---

By examining the statements in Example 12-1, the principles of setting up TLS become more apparent.

**CIPHERSUITE** This specifies the encryption and authentication algorithms that this z/OS FTP server supports. The actual algorithms used are negotiated during the TLS handshake. In this sample, the FTP server will support any of the four combinations listed.

**Tip:** In almost all of the z/OS TCP/IP configuration files, a semi-colon (;) placed anywhere on a line indicates the beginning of a comment.

**SECURE\_FTP** Any FTP client connecting to this FTP server is required to use TLS. Non-TLS connections are rejected. If desired, an FTP server can be configured to have TLS as an option. The choice then becomes the client's.

Flexibility is not necessarily negative in this case—but it does mean that end users with FTP clients may accidentally connect without using TLS.

**SECURE\_LOGIN** This FTP server is configured not to authenticate any FTP clients.

**Note:** TLS authenticates by using a certificate. By default, TLS requires that a server *always* send a certificate to the client. This certificate allows the client to verify that it is really talking to the server it attempted to contact. The idea here is to prevent a computer in the middle from pretending it is the server.

### **SECURE\_CTRLCONN** and **SECURE\_DATACONN**

These two statements control whether or not the server will require the FTP control connection and the FTP data connection to be secure. If an FTP server is being configured for security purposes, it makes sense to make sure that encryption occurs on all communications between the client and the server. Using PRIVATE as the operand means that the FTP server will require TLS for both control and data connections.

A surprising aspect of TLS is that the data connection can be set with an operand of CLEAR, which will allow data to be sent unencrypted.

### **KEYRING**

The key ring is the location of the certification repository. The default certificate in this key ring is the certificate that will be sent out by the FTP server to any clients.

These parameters in the FTP.DATA configuration file control the behavior of the FTP server application. Because the TLS interface is implemented at the top of the transport layer (as shown in Figure 12-1), the FTP server requires specific configuration changes in order to function with TLS. From an application programming perspective, the source code of the FTP server must be modified (by the original developer of the application) to operate using TLS.

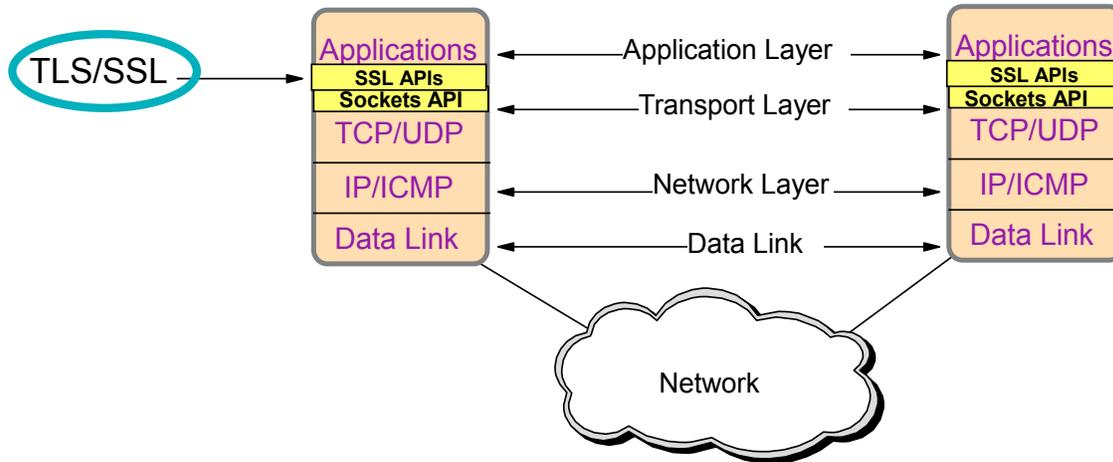


Figure 12-1 TLS in the IP layer model

If the FTP server requires modifications to support TLS, what about the FTP client? The FTP client also requires similar configuration changes. Of special note, however is the certificate requirements.

As noted, the FTP server sends out a *certificate* to prove its identity. How does the FTP client validate this certificate? In the FTP client key ring, there must be a certificate that can vouch for the authenticity of the certificate received from the FTP server. Normally, a Certificate Authority (CA) certificate is used for this purpose.

**Note:** The key ring used by the FTP application is under the control of an external security manager such as RACF. The term *ring* is perhaps a confusing one: it simply refers to a subset of a database. Using RACF commands, a ring is populated with certificates.

## Virtual Private Network

Virtual Private Network (VPN) is a general term used to describe a secure tunnel (data stream) between two endpoints. The term does *not* describe a protocol. The industry standard protocol for a VPN is an architecture called IPSec. The IPSec architecture is outlined in RFC 2401, and its implementation encompasses RFCs 2402, 2406, and 2407 (there are various others, but those are the big three).

There are some similarities between IPSec and TLS. They both provide encryption of data on the network between two endpoints. They both can provide

authentication of those endpoints. An important difference is that IPsec is implemented at the network layer; this is illustrated in Figure 12-2.

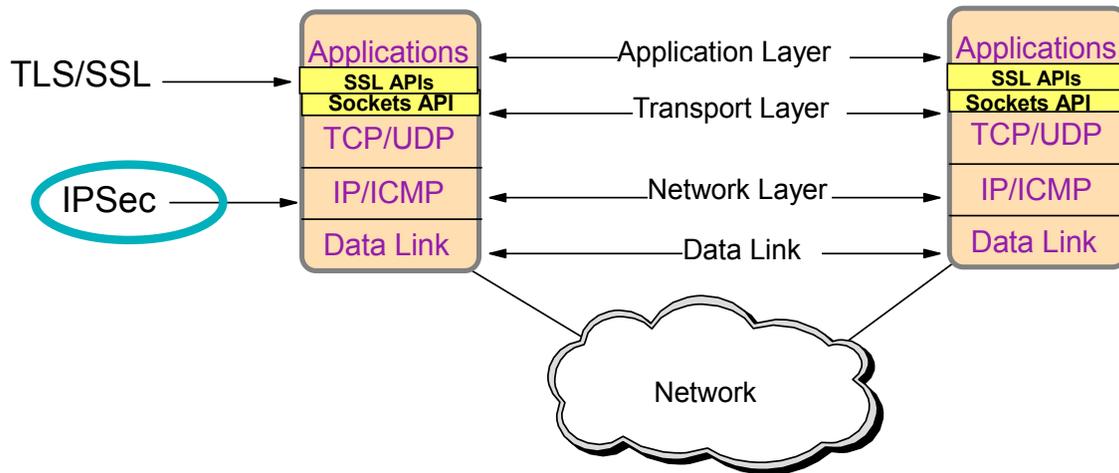


Figure 12-2 IPsec in the IP layer model

Because IPsec is at the network layer, the endpoints of a VPN occur at the TCP/IP stack. The endpoints of TLS occur only at an application (like the FTP server). This implies that the endpoint of a VPN may exist on the same host as the application is running on, or the endpoint could be at an adjacent firewall on the network. It all depends upon the organization's needs.

The other implication of being at the network layer is that all IP traffic can be directed through a VPN. "All traffic" implies not just traffic from different applications, but also traffic from different applications using other protocols like UDP or ICMP. With TLS, only the traffic between the two implementing applications is protected.

A VPN can be further divided into two different types: a manual VPN and a dynamic VPN. Although z/OS supports manual VPNs, they are not very commonly used. Consequently, this text only discusses dynamic VPNs.

A *dynamic* VPN requires a separate server to support the exchange of the keys that will be used to encrypt data at each end point. In z/OS, the key exchange is supported by the IKE daemon. IKE stands for Internet Key Exchange, which is the standard (RFC 2409) protocol used to exchange keys for a VPN.

How is this all accomplished on z/OS? The characteristics of the dynamic VPN are controlled by the TCP/IP stack using information from the policy agent. The policy agent is a daemon that runs with the purpose of reading policy definitions

from a Lightweight Directory Access Protocol (LDAP) server. The policy definitions are in turn read by the TCP/IP stack.

Here's where you can breath a sigh of relief: the policy definitions themselves are created using a graphical user interface application running on a workstation. The application is called the z/OS IP Security Configuration Assistant. It makes the creation of the rules surrounding a VPN a relatively simple task.

## Application Transparent TLS

Application Transparent TLS (AT-TLS) is a unique usage of TLS on the z/OS end of the session. In principle, it is quite simple: Instead of having the application itself be TLS-capable and TLS-aware, the establishment of the TLS connection is pushed down the stack into the TCP layer.

Many applications on z/OS can run without even being aware that the connection is using TLS. Remote clients cannot distinguish between "normal" TLS (where the application is doing the socket calls necessary for TLS) and AT-TLS (where the TCP layer handles the connection).

Figure 12-3 on page 292 shows the AT-TLS layer implemented at a lower layer than the standard TLS. Because TCP/IP is a layered protocol, the changes done at the TCP layer are hidden from the application layer.

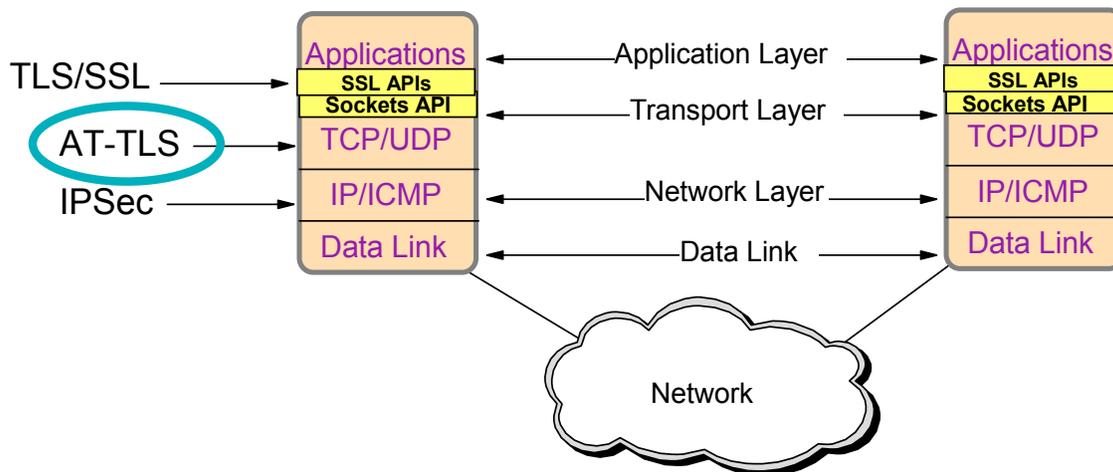


Figure 12-3 AT-TLS in the IP layer model

AT-TLS will appear identical to normal TLS to any application connecting to the z/OS host. The AT-TLS environment is activated by a simple option within the TCPCONFIG statement block in the TCP/IP profile data set: TTLS. When coded,

the TCP/IP stack will use the policy agent (in the same fashion as it does for IPSec) to determine how to handle each application's communication.

**Note:** This is the second time the policy agent has been mentioned as the source of configuration data. So you might wonder, why are some definitions coded in a policy agent and some in specific application configuration files?

The primary reason for this is conceptual: the configuration data that belongs under policy agent control should be information that is related to the policies and goals of the organization. Remember that security choices should flow from a security policy document? The *policy agent* is the service that implements the policies.

The other advantage of the policy agent is that it uses LDAP as the source of the policies. The LDAP directory service is a networked repository of configuration data available to all hosts in the network. A readily apparent advantage is that multiple TCP/IP instances take advantage of policy data stored in LDAP.

Like most protocols relating to IP, LDAP is defined via RFCs: 1777, 2251-2256.

## OpenSSH

OpenSSH stands for Open Secure SHell, and it is sometimes referred to as "secure shell." Unlike TLS and IPSec, OpenSSH is not a formalized standard, but it is used widely in the IP community and consequently it was ported to the z/OS platform. As of the time of writing, the IETF is working on producing RFCs to standardize OpenSSH.

On z/OS, OpenSSH consists of:

- Secure FTP (sftp)** The sftp client and sftp daemons provide secure FTP-like functionality.
- Secure copy (scp)** The scp command is a secure alternative for the remote copy program, rcp.
- Remote login (ssh)** The ssh command functions similarly to the remote login (rlogin) command or remote shell (rsh) commands. The daemon end is supported by the secure shell daemon, sshd.

OpenSSH on z/OS is the same as you would expect to find on any UNIX-like platform.

## 12.3.2 TCP/IP on z/OS security features

There are some security features for which, although implemented on most platforms, no standardization is necessary. Additionally, there are some security features that are completely unique to the z/OS environment. This section discusses both kinds of security features.

### Intrusion Detection Services (IDS)

There are two fundamental varieties of Intrusion Detection Services (IDS). IDS can function within the domain of an individual host, or it can function as a network IDS with a scope including the entire network to which it is attached.

On z/OS, the scope is the former kind only: IDS functions within the z/OS host only and no efforts are made to function outside of the z/OS host. There are specialized platforms designed to perform network IDS, and it does not make sense to use z/OS in such a role.

In z/OS, the IDS capabilities are built into the stack itself. Many of these capabilities are automatically handled by z/OS. For example, malformed packets are automatically discarded, independent of any settings controllable by the system administrator.

But the capabilities of IDS can be expanded to include the following types of incidents:

**Scanning**      A *scan* is a systematic accessing of network resources over a period of time from a single IP address. Scan attacks are not detrimental to a host. However, they are an indication that a host on the network is trying to determine what ports are open for business on the target host. Detection and reporting of scan attacks are important since the host doing the scan may later be the same host that launches a more virulent attack.

**Attacks**        An *attack* on a host can take many forms. It is impossible to list all of them here, but a few examples are flood attacks, redirection attacks, and restricted protocol attacks. IDS can be configured to detect, report, and prevent all well-known attacks.

Intrusion detection is not a precise science. Scans can come in slowly or quickly, depending upon the hacking tool in use. Also, a flood of connection requests may just be a large group of users logging back on to the host. For example, if an intermediate router went down briefly and 5000 users were disconnected—when the router came back up, a flood of new connections could be received at the z/OS host.

Therefore, it is up to an individual organization to determine what sequences of events are to be considered an attack and what sequences are benign. The implementation of the IDS rules is done through the policy agent.

## Using SAF in a network context

What exactly can SAF do in a networking context? It can collectively do more to restrict end-user capabilities than any organization might ever want to implement. In other words, the security features listed here might be used in combinations, but it is unlikely that any organization would want to implement more than a few of these features.

Every organization is unique. Most organizations will find that some of the SAF-based security features listed here have a place in the context of their security policy.

### Stack access

As mentioned in 5.5.2, “The multi-stack environment” on page 100, a single LPAR can contain more than one active TCP/IP stack. And for the most part, if the application has been coded to support it, a stack can be accessed by changing the TCPIPJOBNAME statement in a resolver configuration file. That sounds like a security exposure, and it is.

SAF can be used to restrict which TCP/IP stack can be accessed by any individual user ID.

### Network access

One of the standard methods of penetrating a network is to use intermediate hosts as access points. An organization might want to limit such a possibility. In addition, some user IDs on the z/OS host might be used by either less experienced or less trusted users.

Using SAF, entire networks or subnetworks can be restricted on an individual user ID basis.

### Port access

In order for a hacker to continue through an intermediate host, a port is required.

SAF can be used to restrict the range of ports that a user ID can access. In addition, port access can be used to add extra security to the user IDs associated with a server or daemon. Since some servers may run with z/OS UNIX superuser authority, port access can be used to restrict a user ID to only using the port number(s) it requires.

**Note:** In a UNIX context, *superuser authority* or *root authority* refers to any user ID with a UID of 0. A user ID with a UID of 0 has near complete control over the UNIX operating environment.

On a z/OS host, this is not quite as dangerous as on a pure UNIX-like operating system—but superuser authority still has very serious implications and must be regulated accordingly.

### **Netstat command**

The output from the netstat command, depending upon the option used, can be considered a security risk. For example, information about attached networks or interface type could be used to the advantage of a hacker.

Using SAF, all or some of the netstat command options can be restricted.

**Reminder:** The netstat command on z/OS is available from within a z/OS UNIX or TSO environment. The z/OS netstat command is implemented in the same fashion as the netstat command that is found on Windows and UNIX style operating systems. It can be used to display session and network status of the IP network.

### **Packet tracing**

TCP/IP on z/OS supports a packet tracing service. The service uses a facility of the MVS operating system itself (called CTRACE) to perform low-impact packet tracing, including filtering options. There is a z/OS packet trace formatting environment, but the data can also be exported in formats compatible with workstation trace analysis tools.

Packet tracing should be controlled to prevent unauthorized tracing of data.

So, how does all this SAF control get put into place? Within the SAF environment is a class of resources called SERVAUTH. The SERVAUTH class can have resources defined to represent IP features.

Using the same order as above, the resource definitions begin with something that looks like the following:

- ▶ EZB.STACKACCESS....
- ▶ EZB.NETACCESS....
- ▶ EZB.PORTACCESSSS....
- ▶ EZB.NETSTAT....
- ▶ EZB.NETMGMT....

After such a resource has been defined, a universal access permission must be set. For example, a universal access of `none` might be used. This would implicitly protect the resource from being used by any users. Individual user IDs (or groups of user IDs) could then be given access to the resources as needed.

### More port control

Port control is a big topic in the world of network security. Within an organization, an individual or group responsible for a boundary device firewall might have strict rules as to which ports can be used. And, even within the z/OS host, keeping a tight control on which ports are available is a key to a healthy and secure system.

With respect to a firewall, one of the most difficult areas is ephemeral port usage of an FTP server.

**Reminder:** An *ephemeral port* is one that is required to complete a connection between endpoints, but the actual port number required is unimportant. Ephemeral ports are assigned by the TCP/IP stack in a (roughly) sequential pattern. An application that needs an ephemeral port asks for the port (implicitly), and it is provided by the TCP/IP stack. Ephemeral port numbers are always greater than 1024.

The FTP server may request an ephemeral port for performing a data transfer. By default, this port number could be anywhere in the range from 1024 to 65535. However, it is not desirable to open up such a wide range of ports to a firewall. Using the `PORTRANGE` statement in the `FTP.DATA` configuration file, FTP ephemeral port usage can be limited to a low port and high port range. This same range would be reflected in the appropriate port filtering firewall(s).

Access to ports below 1024 should normally be restricted. This is controlled by the `RESTRICTLOWPORTS` option on the `TCPCONFIG` and `UDPCONFIG` statements in the TCP/IP profile; see Example 12-2.

#### *Example 12-2 Controlling low number ports*

---

```
TCPCONFIG
RESTRICTLOWPORTS
```

```

IPCONFIG
  RESTRICTLOWPORTS
PORT
  20 TCP * NOAUTOLOG           ; FTP Server
  21 TCP FTPD1                 ; FTP Server
  23 TCP TN3270D               ; Telnet 3270 Server
  23 TCP INETD1 BIND 201.2.10.199 ; z/OS UNIX Telnet server
  25 TCP SMTP                  ; SMTP Server
  53 TCP NAMED1                ; Domain Name Server
  53 UDP NAMED1                ; Domain Name Server

```

---

If RESTRICTLOWPORTS has been coded, how can an application such as a Web server or a FTP access their normal ports? The answer is in the PORT statement shown in Example 12-2. A PORT statement entry is required for any application that wants to use a port below 1024 when RESTRICTLOWPORTS is in effect. The TCP/IP stack reserves ports for the started task (server) name listed on the PORT statement.

The PORT statement is a rudimentary form of control: only an application with the assigned task name is allowed to use that port number.

The FTP application has the unusual characteristic of using a second connection for the data transfer. It also allows a user to “hop” to a secondary FTP, a process called *proxy*. Consequently, the FTP environment has some further restrictions possible. Obviously, such controls would be effected using the FTP.DATA configuration data set.

For example, the FTP sub-command PORT can be disabled in a proxy environment by using the PORTCOMMAND statement. Or, the IP address of a PORT or PASV command (see RFC 959) can be forced to match the IP address for the remote end of the control session. This is accomplished using the PORTCOMMANDIPADDR and PASSIVEDATACONN statements, respectively.

## IP filtering

Although the z/OS operating system is not an ideal operating system to run as a routing host, it does have the capability of running fully functional firewall filtering rules. Packets can be permitted or denied based upon source, destination, or port number. Probably the best usage of IP filters on z/OS would be as a secondary line of defense: if a boundary firewall has been compromised, the IP filters on z/OS could form another hurdle to a potential hacker.

In z/OS, IP filtering rules are activated using the policy agent.

## More security options

There are many aspects of the implementation of a network that can be considered a security issue. The following is a short list of some of the features and how they function in a security context:

**Multiple TCP/IP stacks** Although having separate IP stacks within a single LPAR can be compared to having two separate hosts running TCP/IP, it is not quite as secure. However, if a clear delineation between IP endpoints is required within a single LPAR, multiple TCP/IP stacks provides this capability. By using two stacks instead of two IP addresses within a single stack, there is a greater isolation at the two endpoints.

**Socket options** On TCP/IP for z/OS, a SAF resource (EZB.SOCKOPT.\*.\*SO\_BROADCAST) can be activated to prevent applications from activating certain socket options, such as the ability to send broadcast datagrams. This prevents an application from using broadcast datagrams that could flood a network.

## 12.4 TN3270 security

The TN3270 environment is unique and complex enough to warrant some special attention. As mentioned, the TN3270 server supports TLS. In addition, the TN3270 server makes full use of SAF-based authentication. And, if desired, TLS and SAF can be used together to force a TN3270 client to send a certificate that is associated with a SAF controlled user ID, allowing a product like RACF further control.

A sample excerpt of some related TN320 server statements is shown in Example 12-3.

*Example 12-3 TN3270 server security-related statements*

---

```
LUMAP TSOLU001 9.29.168.30 DEFAPPL TSO DEFONLY
MAXREQSESS 40 ;allow max of 40 binds in 10 seconds.
MAXRECEIVE 65535 ;no more than 65KB at a time.
ENCRYPTION
  SSL_DES_SHA ;for testing, keep these cipher
  SSL_RC2_MD5_EX ;cipher statements
  SSL_RC4_MD5_EX ;in this order
  SSL_3DES_SHA ;export restricted?
ENDENCRYPTION
CONNTYPE SECURE ; TLS required
```

CLIENTAUTH NONE  
KEYRING TN3270ring

---

Take a closer look at these statements as they apply to security:

**LU choice** TN3270 configuration statements can control the LU that a given network or IP address can access. In this case, IP address 9.29.168.30 will be assigned a VTAM LU called TSOLU001.

**Application selection**

The application selection can be limited based upon a network or IP address. In Example 12-3 on page 299, the LUMAP statement allows a connection to TSO only.

**TN3270 client behavior**

If a workstation TN3270 client sends too much of a certain type of data, or just too much data at a single time, the connection may be dropped (MAXREQSESS and MAXRECEIVE).

**TLS**

The TLS configuration statements are effectively the same as those for the FTP application.

## 12.5 SNA security

SNA can be roughly divided into two types of implementation: subarea and APPN. The security considerations are slightly different between them.

### 12.5.1 Subarea

The networks that contain genuine SNA traffic are generally not public—or at least are considered to be secure networks, again reducing the security requirements of SNA traffic.

In the event that security measures are considered appropriate for SNA traffic, the following features can be used:

**LU authentication** When using an encrypted session, LU authentication can be performed to certify that the key used by each endpoint is the same. However, if authentication is not requested, the mismatch of the session keys prevents any data from being unencrypted at either end.

**Note:** SNA uses symmetric encryption for LU to LU sessions. This means that the key at each endpoint is the same. The keys must be shared prior to establishing the LU-LU session.

**Message authentication**

An additional code can be sent with all SNA data messages. This code can be used to verify that the message has not been altered in transit.

**Data encryption**

Data between LUs can be encrypted to ensure confidentiality between sessions.

## 12.5.2 APPN security

It is reasonable to state that the majority of APPN traffic is now encapsulated when it is on the network using UDP/IP (that is, using Enterprise Extender, covered in Chapter 9, “SNA/IP implementation” on page 211). In other words, SNA has evolved from being a network architecture. Instead, it is being transformed into a set of protocols that define the architecture for interapplication communications. From an IP standpoint, APPN is an application architecture, not a networking architecture.

When APPN traffic is carried over UDP/IP, standard IP-based security methods can be used, such as VPN tunnels (see Chapter 12, “Network security” on page 283).

For APPN traffic that is not traveling over an IP network, or if IP-based security measures are not considered appropriate or adequate, APPN has the following features available:

**Authentication**

The identity of a session partner can be confirmed by VTAM session level services or at the application program level (user identification).

**Encryption**

An APPN session can be defined to require that data be encrypted between LUs.

## 12.6 Summary

Security entails the reduction of the likelihood of damage, whether intentional or inadvertent. Within a context of z/OS, features such as TLS/SSL, IPsec, and SSH can be used to improve the security of data on the network. Other features such as AT-TLS and SAF-based security can also be used. Intrusion detection

services are already active to some degree within the TCP/IP stack. The policy agent is the repository for many security-related configuration values. Due to the complexity of the SNA architecture and its present limited use in the telecommunications portion of the network, security in this context is not as much of a concern as it is in an IP context.

<b>Key terms in this chapter</b>		
Application Transparent TLS (AT-TLS)	asymmetric encryption	authentication
authorization	IPSec	policy agent
Secure Sockets Layer (SSL)	symmetric encryption	System Authorization Facility (SAF)
Transport Layer Security (TLS)	Virtual Private Network (VPN)	



## Problem determination

**Objectives:** This chapter provides information for diagnosing problems occurring in Communications Server for z/OS (TCP/IP and VTAM). This chapter can help you determine whether a specific problem is a result of the TCP/IP or VTAM implementation.

After completing this chapter, you will be able to:

- ▶ List the different types of system dumps.
- ▶ List the different types VTAM traces.
- ▶ Use z/OS commands to display or modify data and settings.
- ▶ Identify messages associated with each component.

## 13.1 Network problems

Because businesses depend so heavily upon the availability of data processing systems, a problem in the network can be catastrophic. Real money is lost when networks fail. When the network is used for connecting a transaction processing mainframe to the outside world, the losses can be staggering. Keeping the network problem-free and responsive is a priority, so determining where a problem lies and fixing it quickly is imperative.

Your first indication of a network problem may come from users or operators. Users might complain about:

- ▶ An unusual message being received
- ▶ System behavior being different than the past
- ▶ Slow response time on the network
- ▶ No response from the system

System operators might report:

- ▶ Poor performance
- ▶ A TCP/IP or VTAM abnormal end (abend)
- ▶ A loop or wait on the subsystem (TCP/IP or VTAM)

### 13.1.1 Determining the problem

To begin, determine the general cause of the problem by performing these tasks:

- ▶ Read messages in the system log. Each z/OS component has a unique message prefix that identifies the component:
  - VTAM messages are prefixed with IST.
  - TCP/IP messages are prefixed with EZ.
  - CSM messages are prefixed with IVT.
  - TSO messages are prefixed by IKT.

Look for suffix E, which stands for error; for example:

```
IST999E VTAM MESSAGE LOST - INSUFFICIENT STORAGE
```

Check the appropriate messages manual for an explanation of the error.

**Note:** Unsure of the appropriate messages manual? Try the following Web site:

<http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/>

- ▶ Also check the system log to see if the system created an internal memory dump at error time.

- ▶ Find out whether the system has changed; for example, whether the procedures for any components have changed:
  - VTAM procedure
  - VTAM configuration (VTAMLST)
  - TCP/IP procedure
  - TCP/IP profile data set
- ▶ Check whether there has been a recent hardware change to:
  - The central processor complex (CPC)
  - Channel paths
  - OSA-Express (Open System Adapter Express) type or definition
  - Network equipment

After you locate the general problem area, use the tools and diagnostic aids at your disposal to track down the problem.

## 13.2 Tools and diagnostic aids

z/OS has a number of tools and diagnostic aids for VTAM and TCP/IP that you should be aware of.

### 13.2.1 Common (z/OS-wide) tools and diagnostic aids

The following tools and diagnostic aids can be available on, or are utilized by, any z/OS application. Consequently, they are common to both TCP/IP and VTAM.

#### Abnormal end (abend) dump

If an application ends abnormally, it can generate a memory dump for analysis. The dump, known as an *abend dump*, is produced when one of the following occurs:

- ▶ The operator enters a CANCEL command.
- ▶ An abend macro instruction is issued.
- ▶ A job abnormally ends as the result of an invalid operation.

**Note:** So, what is an “invalid operation?” A divide by zero is an obvious example, which could result in an abend 0C9. For more information on abend codes, see *z/OS: MVS System Codes*.

The start procedures for VTAM and TCP/IP usually have special DD statements (with DD names SYSABEND, SYSUDUMP, or SYSMDUMP) that direct the memory dump to a storage device. The resulting dump is written to the data set specified on the DD statement.

## Stand-alone dump

The stand-alone dump might also have been named a last-stand dump. In other words, when an entire z/OS LPAR is in such a state that it is no longer functioning properly, a program called a stand-alone dump program can be executed. The sole purpose of the stand-alone dump program is to move a raw (unformatted) image of the contents of the system's memory.

Examples of states in which a stand-alone dump is taken are when the system is disabled (unable to accept or perform work, referred to as a *disabled wait*), in a loop, or any other condition that significantly impacts system performance.

A stand-alone dump program is effectively its own operating system: z/OS itself is halted and the stand-alone dump program runs on its own. After the dump is completed, z/OS must be re-IPLed. Obviously, stand-alone dumps are rare and disruptive events in a data center. IBM support will be requested to do an analysis of the system to determine what precipitated the failure.

## SVC dump

SVC dumps contain a summary dump, control blocks, and other system code, but the exact areas that are dumped depends on system settings.

**Reminder:** An SVC is a supervisor call, which effectively executes a system macro. The hexadecimal operation value of an SVC is X'0A'. The SVC number assigned to an abend is X'0D'.

If a program wants to execute an SVC dump (it can also be done through an operator at the z/OS console), the instruction used is X'0A0D'.

An SVC dump is written to a SYS1.DUMPnn data set (if allocated), the SYSMDUMP output data set, or the data set specified on the DCB operand of the SDUMP macroinstruction.

SVC dumps are produced when:

- ▶ A program exception occurs. VTAM or TCP/IP may or may not be terminated as part of this process.
- ▶ An operator requests a dump with the operating system DUMP command.
- ▶ An operator uses a SLIP command with ACTION=SVCD specified, and an event occurs that matches the trap indicated in the SLIP.
- ▶ A macro instruction issues an abend, and there is a DD statement with DDNAME=SYSMDUMP.
- ▶ An SDUMP macro instruction is issued.

- ▶ System recovery routines produce an SVC dump due to a VTAM or TCP/IP error.

### **Interactive Problem Control System (IPCS)**

Dumps, although undesirable, do happen. When they occur, the obvious goal is to capture (or dump) information that will assist in problem determination and ultimately avoid having the problem recur. The writing of the dump to a tape or disk device must be done as efficiently as possible. Consequently, all dumps (except the rarely used SYSABEND and SYSUDUMP) are written unformatted.

IPCS is the tool used to format (for easier reading) a dump of any address space running on z/OS. Control blocks can be formatted by IPCS, as well summaries and analyses of many aspects of the execution environment. IPCS can also be used to format VTAM and TCP/IP traces.

### **Additional documentation for VTAM and TCP/IP problems**

In addition to the collected data just mentioned, system operators should answer the following questions as part of problem source identification.

Presuming that VTAM, TCP/IP, and the network were running well at some point, the question then becomes: what has changed? Obviously, initial problem source identification should focus on the area that has been changed.

Other useful information includes:

**PTFs involved**      A PTF (program temporary fix) could have a problem, or the process of applying the PTF might not have been completed correctly. Search the IBM customer support data base using the PTF number. PTF numbers are only useful when the actual failing load module or component is known. Otherwise, the approximate maintenance level can be determined by the maintenance level of the entire system, usually referred to as a PUT level (PUT stands for program update tape).

**Note:** PTF stands for program temporary fix, which is not necessarily a good description of what a PTF really is. A PTF is a software program fix and it is usually quite permanent. A PTF is written by IBM development and change teams. It is made publicly available for customers to download and apply to their system if desired.

A PTF is described by a document called an APAR, or authorized program analysis report. An APAR describes the symptoms and what was fixed or changed to correct the problem. The APAR contains a pointer to one or more PTFs.

#### **Device types involved!**

If the problem is associated with the use of a particular type of terminal or other hardware unit, the device type should be included in the diagnostic information.

#### **Application programs involved**

Sometimes the problem is associated with a VTAM application program that is an IBM licensed program (such as CICS, IMS or TSO).

#### **Hardware error conditions**

Sometimes it is immediately apparent that a problem is related to a specific hardware error condition. If a hardware error occurred, note the failure condition that accompanied it, such as UNIT CHECK or TIMEOUT.

#### **Coding changes**

A problem can occur after you make coding changes; for example, TCP/IP or VTAM network definitions or VTAM macro changes. Changes to applications and exits are included as potential problem points.

### **13.2.2 VTAM tools and diagnostic aids**

The following tools and diagnostic aids are related to VTAM only.

#### **First Failure Support Technology (FFST) dump**

First Failure Support Technology™ is a licensed program that captures information about a potential problem event when it occurs. Based on options specified, you may get a dump and an alert. This helps you, particularly with intermittent problems, because the problem is captured the first time it happens, so you do not have to wait for the problem to occur a second time.

You may use the dump formatting CLIST FFSTDF to format your VTAM FFST™ minidump. FFSTDF formats your minidump and writes it to a data set, which you can view online or print using the IEBTPCH utility program.

## VTAM CSDUMP

VTAM has the ability to capture a dump using its own MODIFY *procname*,CSDUMP command. The dump can be captured immediately, or it can be set up to dump in response to VTAM messages or sense codes.

## Network traces

You can activate VTAM traces when you start VTAM by using the TRACE option on the START command, or you can activate traces when VTAM is already running by using the MODIFY TRACE command.

## Sense codes

When an unexpected condition occurs, VTAM sets a sense code to provide diagnostic (though sometimes difficult to interpret) information. To assist in problem determination, VTAM hints have been added to many of the architected sense code definitions.

See Example 13-1 for a sample of decoding a VTAM sense code.

*Example 13-1 Decoding 08010010 (Request Reject)*

---

### 08010010

08 Request reject

**01** Resource not available: The LU, PU, link station, or link specified in an RU is not available.

**0010** A switched subarea connection cannot be established because no switched subarea links are defined.

---

VTAM sense codes are four bytes long, as explained here:

- ▶ The first two bytes together are referred to as the *sense code*.
  - Breaking down these two bytes, the first (x'08' in Example 13-1) defines the category.
  - The second byte (x'01) modifies this category.
- ▶ The final two bytes (x'0010') are specific to the sense code and may contain user-defined or application data.

For additional information on sense codes, refer to *z/OS Communications Server: IP and SNA Codes*.

### **VTAM internal trace**

The VTAM internal trace (VIT) provides a record of the sequence of events within VTAM. It is probably the most commonly used VTAM trace option.

These internal events include the scheduling of processes, the management of storage, and the flow of internal path information units between VTAM components.

Both the TRACE start option and the MODIFY TRACE command have an OPTION operand you can use to select VIT options.

### **Other VTAM traces**

Several other VTAM trace options exist:

- ▶ The buffer contents trace shows the contents of inbound and outbound message buffers.
- ▶ The I/O trace shows all I/O sent between VTAM and a particular network node.
- ▶ The SMS (storage management services) trace shows information about the use of buffers, including how often a buffer pool has expanded, how many buffers are currently being used, and what was the maximum number of buffers used since the last trace record was written.
- ▶ The TGET/TPUT trace shows each message as it passes between a TSO command processor and TSO/VTAM.

### **VTAM DISPLAY commands**

From SDSF or the z/OS console, you can issue various commands to display aspects of VTAM. For example, some of the more common commands are:

D NET,APPLS	Provides status of application program major and minor nodes
D NET,BFRUSE	Shows VTAM buffer usage
D NET,MAJNODES	Provides status of major nodes
D NET,ROUTE	Provides status of explicit routes and virtual routes, the existence of routes, and whether a route is operational
D NET,TRL	Provides information about the TRL major node or about a single TRLE definition statement
D NET,VTAMOPTS	Provides information about the VTAM start options

**Reminder:** TRL stands for transport resource list. A TRL is a VTAM major node that can be defined on a VBUILD statement, or it may be dynamically created. The major node consists of TRL elements, or TRLEs. A TRLE defines the communications characteristics of multipath channel (MPC) or dynamic XCF connections.

MPC is commonly used for the OSA-Express adapter.

The following are some sample outputs from VTAM display commands.

D NET,BFRUSE

Example 13-2 shows the command output.

*Example 13-2 D NET,BFRUSE output*

---

```
DISPLAY ACCEPTED
DISPLAY TYPE = BUFFER POOL DATA 617
I000   BUFF SIZE  515           EXP INCREMENT  14
        TIMES EXP  0           EXP/CONT THRESH 48
        CURR TOTAL 182         CURR AVAILABLE 182
        MAX TOTAL  182         MAX USED          13
        EXP LIMIT 211155       BUFFS REQUESTED 0
-----
BS00   BUFF SIZE  260           EXP INCREMENT  14
        TIMES EXP  0           EXP/CONT THRESH 14
        CURR TOTAL 28         CURR AVAILABLE 28
        MAX TOTAL  28         MAX USED           0
-----
LP00   BUFF SIZE 2032           EXP INCREMENT   6
        TIMES EXP  1           EXP/CONT THRESH  1
        CURR TOTAL 16         CURR AVAILABLE 14
        MAX TOTAL  16         MAX USED           11
-----
XD00   BUFF SIZE  697           EXP INCREMENT   5
        TIMES EXP  5           EXP/CONT THRESH  4
        CURR TOTAL 10         CURR AVAILABLE 10
        MAX TOTAL  30         MAX USED           21
-----
CSALIMIT = 120661K, CURRENT = 6755K, MAXIMUM = 6787K
MAXIMUM CSA USED = 6787K
SYSTEM CSA LIMIT = 134068K
65% OF SYSTEM CSA STORAGE REMAINING = 87867K
CSA24 LIMIT = NOLIMIT, CURRENT = 51K, MAXIMUM = 53K
MAXIMUM CSA24 USED = 53K
```

```
IRNLIMIT = NOLIMIT, CURRENT = OK, MAXIMUM = OK
VTAM PRIVATE: CURRENT = 1272K, MAXIMUM USED = 1338K
-----
```

---

D NET,TRACES

Example 13-3 shows the command output.

*Example 13-3 D NET,TRACES output*

---

```
DISPLAY ACCEPTED
DISPLAY TYPE = TRACES,TYPE=VTAM
VTAM INTERNAL TRACE ACTIVE - MODE = INT, SIZE = 999 PAGES
OPTIONS = PSS SMS
END
```

---

In this display, you can see that the VTAM internal trace is running and active. The VTAM internal trace options active are PSS and SMS (buffer use).

PSS stands for process scheduling services, which is the component of VTAM that handles the scheduling and dispatching of VTAM routines.

### **The generalized trace facility**

VTAM passes all external trace data to the generalized trace facility (GTF). GTF must be active to use VTAM traces. Consequently, trace options must be specified at the starting of the GTF procedure.

Example 13-4 on page 312 is a sample start procedure.

This procedure must reside in SYS1.PROCLIB. The external trace file produced is by default named SYS1.TRACE.

*Example 13-4 GTF procedure*

---

```
//GTFNEW PROC MEMBER=GTFPARM
//IEFPROC EXEC PGM= AHLGTF, PARM='MODE=EXT,DEBUG=NO,TIME=YES',
// TIME=1440,REGION=2880K
//IEFRDER DD DSN=SYS1.TRACE,UNIT=SYSDA,SPACE=(TRK,20),
// DISP=(NEW,KEEP)
//SYSLIB DD DSN=SYS1.PARMLIB(&MEMBER),DISP=SHR
```

---

In library SYS1.PARMLIB, the member GTFPARM should contain the VTAM trace options. For example:

TRACE=SYSM,USRP,TRC,DSP,PCI,SRM

Table 13-1 lists the VTAM trace components that can be used. All of these VTAM trace components are captured under the GTF trace option USR.

In other words, GTF must be tracing USR in order to capture the VTAM component records as outlined in Table 13-1. By specifying USRP in SYS1.PARMLIB(GTFPARM), the operator receives a prompt at GTF startup. At the prompt, the specific VTAM components can be entered as desired.

Table 13-1 VTAM trace component

Number	Trace
FE1	VTAM internal trace
FEF	VTAM buffer contents trace (TSC component) Trace output says "VTAM."
FF1	SME buffer trace and VTAM buffer contents trace (API component) Trace output says "USER."
FF0	SMS (buffer use) trace
FF2	NCP 37xx line or TG trace
FE2	TSO/VTAM TGET/TPUT trace
FE4	Line PIU, generalized PIU, or network controller line trace

You can use the Interactive Problem Control System (IPCS) to analyze traces written to the GTF output data set.

### 13.2.3 TCP/IP tools and diagnostic aids

In addition to abnormal end dumps, TCP/IP has various tools and diagnostic aids for problem source identification.

**Note:** TCP/IP utilizes VTAM for both memory (Communications Storage Manager, CSM) services and for communications I/O. Consequently, when analyzing TCP/IP problems, VTAM diagnostic data may also be required.

#### TCP/IP component traces

By utilizing the z/OS component trace facility, you can store trace data efficiently (with low system impact) into data sets. After tracing of a failure is complete, IPCS can be used to format and view the data.

The components of TCP/IP that can be traced are:

- ▶ SYSTCPDA for IP packet tracing
- ▶ SYSTCPIP for TCP/IP internal event tracing
- ▶ SYSTCPRT for OMPROUTE tracing
- ▶ SYSTCPRE for TCP/IP resolver tracing
- ▶ SYSTCPIS for the TCP/IP intrusion detection services trace

By far, the SYSTCPDA packet trace is the most commonly used component trace. The other trace options are usually utilized only as a result of instructions from IBM support personnel or by advanced TCP/IP system administrators. Consequently, they are not discussed further in this text.

Component trace writes trace data either to memory (the TCP/IP dataspace) or to an external writer program, which writes the data to storage. The following command sequence activates a SYSTCPDA packet trace and stores the unformatted data:

1. Start the writer for packet tracing:

```
TRACE CT,WTRSTART=pktwrt
```

where *pktwrt* represents the JCL library member name that is used to invoke the external writer program.

**Note:** When reading a command like the above, the upper case portion represents the actual command itself, while anything in lower case represents a user-supplied parameter. This convention is used throughout z/OS manuals.

2. Clear any previous packet trace settings:

```
V TCPIP,tcpproc,PKTTRACE,CLEAR
```

where *tcpproc* identifies the TCP/IP address space.

3. Start TCPIP packet trace:

```
V TCPIP,tcpproc,PKTTRACE,ON,IPADDR=nn.nn.nn.nn
```

The IP address of the client, *nn.nn.nn.nn*, was used to filter the packet trace entries to be captured. There are a large number of filters available for a packet trace that are not shown here.

4. Start packet trace and connect the writer:

```
TRACE CT,ON,COMP=SYSTCPDA,SUB=(tcpproc)  
R xx,WTR=pktwrt,END
```

where *xx* is an identifier from the TRACE prompt that asks for TRACE options.

5. Recreate the problem.
6. STOP all traces as soon as the problem occurs so that the trace entries do not wrap (component traces write continuously, starting over at the beginning of a data set after it is full).

```
TRACE CT,OFF,COMP=SYSTCPDA,SUB=(tcpproc)
TRACE CT,WTRSTOP=pktwrt
```

**Note:** How do you take a dump from the z/OS console? For TCP/IP, try the following sequence of commands. Note that this includes a dump of the TCP/IP data space, which can contain some internal component trace records useful for diagnosis:

```
DUMP COMM=(dump title)
R xx,SDATA=(CSA,SQA,RGN,TRT,GRSQ),CONT
R xx,JOBNAME=(tcpproc),CONT
R xx,DSPNAME=('tcpproc'.TCPIPDS1),CONT
R xx,END
```

As mentioned, for efficiency, component traces are written without any formatting. IPCS can be used to format component trace entries. In particular, packet traces can be formatted to provide statistical and summary information with respect to any traced connection.

### Diagnostic TCP/IP commands

There are also various commands that can be used to display TCP/IP and network information. The following commands can be issued from an ISPF/TSO session. By omitting the TSO prefix, the same commands can be entered at a UNIX System Services prompt. However, the command must be in lower case. The UNIX command options for Netstat are shown in brackets.

TSO NETSTAT ROUTE	Displays routing information (-r).
TSO NETSTAT HOME	Displays the home IP addresses for the IP stack (-h).
TSO NETSTAT DEV	Displays the device status for the defined network interfaces (-d).
TSO NETSTAT STATS	Displays performance statistics (-S).
TSO NETSTAT VIPADCFG	Displays dynamic VIPA configuration data (-F).
TSO NETSTAT SOCKETS	Displays sockets based on client name (-s).
TSO NETSTAT CONN	Displays information for all TCP/IP connections (-c).

Alternatively, these commands can be issued at the z/OS console using a syntax such as:

```
DISPLAY TCPIP,tcpproc,NETSTAT,ROUTE
```

Other commands that are available for network diagnostics (either from a ISPF/TSO environment or from UNIX System Services) are:

**PING** This command can be very useful in determining whether IP connectivity exists. Excellent options are LENGTH and COUNT:

- With LENGTH, you can try various packet sizes.
- With COUNT, you can determine the number of times to execute the ping.

Note that this command uses ICMP ECHO packets—and some routers may be configured to not respond to ICMP ECHO requests. For this reason, a ping can fail in some instances when, for instance, a TCP connection might work fine.

**TRACERTE** With this command you can find the last router a packet can reach in a disrupted network, or you can verify if packets flow over a planned route.

Example 13-5 illustrates a sample of these commands placed into a batch job. The batch output can be kept for further analysis.

*Example 13-5 Statistics JCL*

---

```
//NETSTAT JOB 1,REGION=0M
//NETSTAT EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSPRINT DD DSN=h1q.NETSTAT.OUTPUT,DISP=SHR
//SYSTEM DD DUMMY
//SYSIN DD *
NETSTAT CONFIG
NETSTAT DEV
NETSTAT ROUTE
NETSTAT STATS
NETSTAT BYTE
NETSTAT SOCKETS
NETSTAT CLIENT
PING 127.0.0.1
/*
```

---

Notice the program executed in this sample, IKJEFT01. IKJEFT01 executes in TSO, which means this NETSTAT command is really running in a batch TSO environment.

Example 13-6 shows NETSTAT ROUTE command output.

*Example 13-6 NETSTAT ROUTE output*

Destination	Gateway	Flags	Refcnt	Interface
-----	-----	-----	-----	-----
Default	9.12.4.92	<b>UGS</b>	000001	OSA23AOLNK
9.12.4.0/23	0.0.0.0	US	000000	OSA23AOLNK
9.12.4.20/32	0.0.0.0	<b>UH</b>	000000	OSA23AOLNK
9.12.4.21/32	0.0.0.0	UH	000000	STAVIPA1LNK
10.1.100.0/24	0.0.0.0	US	000000	IQDIOLNK0A016404
10.1.100.4/32	0.0.0.0	H	000000	EZASAMEMVS
10.1.100.4/32	0.0.0.0	UH	000000	IQDIOLNK0A016404

Following is an explanation of the flags in the NETSTAT output.

**Flag U** This flag indicates that the route entry is up and running or ACTIVE. If there is no U, then the route entry is defined but not active. This may be because the device is in a NOT ACTIVE status.

**Flag G** This flag indicates that the route entry specifies an indirect route. That means the destination indicated on the route entry is behind a router from this z/OS system. If there is no G, then the route entry specifies a direct route. That means the destination indicated on the route entry is on the same local network.

**Flag H** This flag indicates that the destination field in this route entry specifies a host route. That means this route is used only if the destination IP address of a datagram exactly matches all 32 bits (255.255.255.255) in the route entry destination field.

If there is no H, then the destination field in this route entry specifies a network route. That means this route is used only if the destination IP address of a datagram exactly matches all the network bits (less than 32 bits, for example, 255.255.255.0) in the route entry destination field.

**Flag S** Indicates the route is a static route that cannot be replaced by a routing daemon (such as OMPROUTE).

Example 13-7 shows the command output from a NETSTAT HOME command.

*Example 13-7 NETSTAT HOME output*

Home address list:

Address	Link	Flg
-----	----	---
9.12.4.20	OSA23A0LNK	P
9.12.4.21	STAVIPA1LNK	
10.1.100.4	EZASAMEMVS	
10.1.100.4	IQDIOLNK0A016404	
10.1.100.4	EZAXCF47	
10.1.100.4	EZAXCF48	
10.1.100.4	EZAXCF52	
10.1.100.4	EZAXCF49	
10.1.100.4	EZAXCF54	
10.1.100.4	EZAXCF55	
10.1.100.4	EZAXCF69	
127.0.0.1	LOOPBACK	

Flag **P** This flag indicates the link OSA23A0LNK is the primary interface. The primary interface can be significant for any applications that make a request to TCP/IP asking for the default IP address. This address is not used for routing purposes.

The interfaces beginning with EZAXCF all have the same IP address and represent various dynamic XCF links to other TCP/IP stacks running on other LPARs within the sysplex.

### 13.3 VTAM problem determination

Table 13-2 lists sample problems that may occur in VTAM. The appropriate actions are explained in Table 13-3 on page 319.

*Table 13-2 VTAM problem types and appropriate actions*

Symptom	Problem type	Action
abend message	abend	Perform <b>A</b>
Activating network nodes takes too long	Performance	Perform <b>B</b>
Application program reports an unexpected return or sense code	Incorrect output or message	Perform <b>C</b>
Deactivating network nodes takes too long	Performance	Perform <b>B</b>
Error message	Message	Perform <b>C</b>

Hung session, LU, or terminal	Incorrect output	Perform <b>D</b>
Hung system	Wait	Perform <b>E</b>
IST error message	Message	Perform <b>C</b>
LOGON takes too long to complete	Performance	Perform <b>B</b>
Performance is degraded after a network outage	Performance	Perform <b>B</b>
Response time is slow	Performance	Perform <b>B</b>
Storage message IST154I or IST561I-IST566I	Storage	Perform <b>F</b>

## ACTIONS

Table 13-3 lists and describes the appropriate actions to perform for further diagnostic information.

*Table 13-3 Appropriate actions*

Action	Description
<b>A</b>	Collect the following documentation: <ul style="list-style-type: none"> <li>▶ Dump output</li> <li>▶ LOGREC</li> <li>▶ Symptom string</li> <li>▶ Abend or system completion code</li> <li>▶ Contents of the general registers (at the time of the abend)</li> <li>▶ PSW (at the time of the abend)</li> <li>▶ VTAM internal trace (at the time of the abend)</li> </ul> <b>Note:</b> LOGREC is the log recording data set used by z/OS. Any abend that occurs is recorded here. In addition, hardware failures are reported in LOGREC.
<b>B</b>	Collect the following documentation: <ul style="list-style-type: none"> <li>▶ System console log</li> <li>▶ Error file output in LOGREC</li> <li>▶ SMS (buffer use) trace output</li> </ul>
<b>C</b>	Collect the following documentation: <ul style="list-style-type: none"> <li>▶ Issuing module</li> <li>▶ Message number</li> <li>▶ System console log</li> <li>▶ VTAM internal trace output</li> </ul>

Action	Description
D	Collect the following documentation: <ul style="list-style-type: none"> <li>▶ VTAM internal trace output (all except LOCK)</li> <li>▶ Generalized trace facility CCWTRACE output</li> </ul>
E	Collect the following documentation: <ul style="list-style-type: none"> <li>▶ I/O trace output</li> <li>▶ Buffer contents trace output</li> <li>▶ Dump of the VTAM primary address space including the common service area (CSA)</li> <li>▶ Output from the VTAM internal trace (all options except LOCK)</li> </ul>
F	Collect the following documentation: <ul style="list-style-type: none"> <li>▶ Full VTAM dump</li> </ul>

## 13.4 TCP/IP problem determination

If TCP/IP abends, as noted in 13.2.1, “Common (z/OS-wide) tools and diagnostic aids” on page 305, a dump should be produced. What you discover in the dump directs your search for the problem source. However, most TCP/IP problems have more subtle symptoms. TCP/IP problems can produce many different symptoms, particularly in a load balancing and sysplex environment. Within this text, the focus remains on the more basic problems that could be encountered.

In most instances, IP problems are reported as one of the following:

- Connectivity problems** The target host cannot be contacted over the network.
- Response time problems** The host is not responding in a timely fashion.
- Performance problems** The data is not moving at the desired or expected rate. This is also called a *throughput* problem and is usually associated with bulk data transfer.

The difficult part is knowing the source of such a problem. For example, is the problem a result of the TCP/IP address space, the target application, the target host itself, or is it an intermediate host (router) somewhere in between? Could it be switching equipment? Or could the problem itself begin at the workstation or host that is attempting the connection?

The following suggestions can help you to narrow down the source of the problem.

## 13.4.1 Messages

The primary location to check for messages, whether issued by TCP/IP or an IP application, is the z/OS system log. Messages might also appear in the SYSPRINT, SYSERR, SYSERROR, and SYSDEBUG data sets. DD statements for these data sets are usually configured to direct messages to the joblog—but check your JCL, to be sure. As noted earlier, TCP/IP messages begin with the EZ prefix. The z/OS Communications Server IP Messages manuals (all four volumes) contain excellent details on why a message may have been issued.

**Note:** The standard TCP/IP applications that are included in z/OS Communications Server also issue EZ prefixed messages. FTP, TN3270, telnet, SMTP and many other applications write EZ messages that can be used for diagnostic information.

Messages might tell exactly what the problem is, or perhaps they might at least direct the system programmer to where to focus attention. A sample TCP/IP message is shown in Example 13-8.

*Example 13-8 TCP/IP TN3270 message*

---

```
EZZ6035I TELNET DEBUG CLIENT IPADDR..PORT
168.214.219.106..4319
CONN: 0000A427 LU: TESTTRLB MOD: EZBTTRCV
RCODE: 1001-01 Client disconnected from the connection.
PARM1: 00000000 PARM2: 00000000 PARM3: 00000000
```

---

This message is actually issued by the TN3270 server application and is a straightforward indication that a TN3270 client has disconnected from the TN3270 server.

## 13.4.2 TCP/IP server or client address space problems

Sometimes the problem manifests itself when a TCP/IP server or client address space (application) stops processing. Or perhaps the application is looping or in a slowdown. The following actions would be appropriate in such a situation:

1. Obtain an SVC dump of TCP/IP or the looping TCP/IP application by issuing the DUMP command from the z/OS system console. If the loop is disabled, the z/OS system console is not available for input so take a stand-alone dump.
2. If the application itself issued any error codes or messages, keep them available because sometimes these messages contain return or reason code details that are system-related rather than application-related.

3. Obtain the appropriate portion of the z/OS system console log.
4. Obtain the job log from the started procedure.
5. Obtain the LOGREC output.

### 13.4.3 Diagnosing network problems

The following basic sequence can assist in diagnosing a network-based problem:

1. Test and verify the TCP/IP address space configuration using NETSTAT commands.
2. Test connectivity to remote hosts using the PING and TRACERTE commands.
3. Obtain a TCP/IP packet trace (component SYSTCPDA).

The packet trace can be especially useful for determining where delays or response failures occur. By examining time stamps, you can determine whether a delay is at the z/OS end of the connection or somewhere else on the network.

#### Mainframe packet trace

The component SYSTCPDA trace is one of the starting points for diagnosing IP-based problems. The trace is written unformatted to a CTRACE data set, and the data can subsequently be formatted using IPCS. There are many filtering and report generation options available. One of the most commonly used report options is a connection summary option called SESSION(DETAIL). A sample is shown in Example 13-9.

*Example 13-9 Excerpt of packet trace session output*

---

Local Ip Address:	198.162.245.166
Remote Ip Address:	142.178.32.65
Host:	Local, Remote
Client or Server:	SERVER, CLIENT
Port:	21, 3930
Application:	ftp,
Link speed (parm):	10, 10 Megabits/s
Connection:	
First timestamp:	2005/09/20 09:47:25.592108
Last timestamp:	2005/09/20 09:47:25.808723
Duration:	00:00:00.216615
Average Round-Trip-Time:	0.019 sec
Final Round-Trip-Time:	0.312 sec

```

Final state:                                CLOSED (PASSIVE CLOSE)
Out-of-order timestamps:                    0

Data Quantity & Throughput:
Inbound,                                     Outbound
Application data bytes:                      130,                      123
Sequence number delta:                      132,                      124
Total bytes Sent:                           130,                      123
Bytes retransmitted:                         0,                        0
Throughput:                                  0.75,                      0.709 Kilobytes/s
Bandwidth utilization:                       0.06%,                     0.05%
Delay ACK Threshold:                         200,                       200 ms
Minimum Ack Time:                           0.038900,                  0.000000
Average Ack Time:                           0.039707,                  0.000000
Maximum Ack Time:                           0.040515,                  0.000000

Data Segment Stats:
Inbound,                                     Outbound
Number of data segments:                     1,                          2
Maximum segment size:                       4056,                       1460
Largest segment size:                       130,                        63
Average segment size:                       130,                        61
Smallest segment size:                      130,                        60
Segments/window:                            1.0,                        1.0
Average bytes/window:                       130,                        61
Most bytes/window:                          130,                        63

Window Stats:
Inbound,                                     Outbound
Number of windows:                           1,                          2
Maximum window size:                         65536,                      65536
Largest window advertised:                   65535,                      65406
Average window advertised:                   52415,                      54526
Smallest window advertised:                   0,                          32768
Window scale factor:                         0,                          0
Window frequency:                            0.0001,                     0.0001 Windows/s

```

---

The SESSION(DETAIL) output gives an at-a-glance summary of the connection that was traced. It begins with the basics, such as IP addresses and port numbers, and then continues on with all other measurable aspects of a TCP/IP connection. The connection in Example 13-9 on page 322 was of very short duration and only a few bytes were exchanged. Considering that it is connected to port 21 on the mainframe, the bytes exchanged would presumably be FTP commands flowing along the control connection of an FTP session.

Much of the other information in a packet trace requires sophisticated knowledge of the TCP and IP protocols.

## LAN-based tracing

When the problem appears to be outside the mainframe, a sniffer (LAN) trace may be appropriate. A sniffer trace may be used to run a trace at the remote end of the connection at the same time as a packet trace is running on the mainframe. By comparing the two traces, the location of the problem can be more accurately pinpointed. For example, by comparing timestamps, a response time problem can be confirmed as being on the mainframe, the remote host, or on the network in between.

There are many tools that perform packet sniffing, network monitoring, and protocol analyzing. Two of the most common are the SNIFFER Tool from Network General, or ETHEREAL which is an Open Source Software released under the GNU General Public License.

## 13.5 Communications Storage Manager (CSM)

Communications Storage Manager (CSM) is a component of VTAM that enables host applications to share data with VTAM and other CSM users without having to physically copy the data. CSM reduces CPU utilization and optimizes system performance during the transfer of bulk data by enabling applications to share buffers.

The code for CSM comes with VTAM, but after CSM is started it runs independently of VTAM and can be used by other z/OS tasks and subsystems.

CSM is started automatically when it is first invoked (by an application requesting the creation of a pool of buffers), and continues to run even if VTAM terminates. While CSM is not in use it retains a minimum amount of storage. It terminates only when z/OS itself terminates.

### 13.5.1 Debugging CSM

CSM problems generally manifest themselves as a central storage problem: either CSM has consumed too much storage, or else an application cannot get any more storage from CSM. The following sequence of instructions provide a good diagnostic approach to CSM problems.

1. Issue the following commands to determine how much storage is being used and by what job name or address space ID (ASID):

```
D NET,CSM,OWNERID=ALL  
D NET,CSM
```

2. Activate CSM VTAM traces:

```
MODIFY NET,TRACE,TYPE=VTAM,
```

```
OPTION=(CSM,CIO,SMS,PSS,PIU,MSG,SSCP,API,NRM,XBUF,APPC,CIA)
```

3. If the DISPLAY CSM output indicates TCP/IP owns the storage:
  - a. Update CTIEZB00 (the TCP/IP internal component trace settings) in the system PARMLIB to specify BUFSIZE(16M). This provides a 16 M CTRACE buffer. TCP/IP must be recycled after this change.
  - b. Restart TCP/IP and issue:

```
TRACE CT,ON,COMP=SYSTCPIP,SUB=(tcPIP)  
R xx,OPTIONS=(STORAGE,VTAM,VTAMDATA,MESSAGE,INTERNET,TCP),END
```

Note that *xx* is the reply number for the response message listed on the console.
4. Obtain console dumps VTAM and TCP/IP.
5. If display output indicates that CSM storage growth is in one of the CSM dataspace pools, dump the dataspace named in the D net,CSM output in addition to the dump of VTAM and TCP/IP.

To dump CSM dataspace:

```
DUMP COMM=(CMS Dataspace Dump)  
R xx,JOBNAME=(net),DSPNAME=(1.dddddd),  
STOR=(00000000,7FFFFFFF),SDATA=(NUC,CSA,RGN,LSQA,SQA,TRT)
```

Note that *ddddddd* is the name of the dataspace.

6. If the DISPLAY CSM output indicates TCP/IP owns the storage, also dump the TCP/IP dataspace because you turned on CTRACE in step 3:

```
R xx,SDATA=(CSA,SQA,RGN,TRT,NUC),JOBNAME=(tcPIP),CONT  
R yy,DSPNAME=(1.*CSM,'tcPIP'.*)
```

## Monitoring CSM

CSM and VTAM buffer usage can be monitored using the following commands.

```
D NET,CSM  
D NET,CSM,OWNERID=ALL
```

You can use the following display commands for monitoring storage:

```
D NET,BFRUSE,BUFFER=SHORT  
D NET,STORUSE
```

Example 13-10 shows the command output of a DISPLAY *net*,CSM command.

*Example 13-10 D NET,CSM,OWNERID=ALL output*

---

```
IVT5549I PROCESSING DISPLAY CSM COMMAND - OWNERID SPECIFIED  
IVT5530I BUFFER BUFFER  
IVT5551I SIZE SOURCE STORAGE ALLOCATED TO OWNER  
IVT5532I -----
```

```

IVT5553I    4K  ECSA                                84K
IVT5554I TOTAL  ECSA                                84K
IVT5532I -----
IVT5553I    4K  DATA SPACE 64                      28K
IVT5554I TOTAL  DATA SPACE 64                      28K
IVT5532I -----
IVT5554I TOTAL  DATA SPACE                          28K
IVT5532I -----
IVT5556I TOTAL  FOR OWNERID                          112K
IVT5557I OWNERID: ASID = 0069  JOBNAME = VTAM
IVT5532I -----
IVT5530I BUFFER BUFFER
IVT5551I SIZE  SOURCE                                STORAGE ALLOCATED TO OWNER
IVT5532I -----
IVT5553I    4K  ECSA                                280K
IVT5553I   16K  ECSA                                32K
IVT5553I   32K  ECSA                                96K
IVT5554I TOTAL  ECSA                                408K
IVT5532I -----
IVT5553I    4K  DATA SPACE 64                      8692K
IVT5553I   16K  DATA SPACE 64                       48K
IVT5553I   32K  DATA SPACE 64                       96K
IVT5554I TOTAL  DATA SPACE 64                      8836K
IVT5532I -----
IVT5554I TOTAL  DATA SPACE                          8836K
IVT5532I -----
IVT5556I TOTAL  FOR OWNERID                          9244K
IVT5557I OWNERID: ASID = 0075  JOBNAME = TCPIP
IVT5599I END

```

---

In this example, job name TCPIP has 9.244 MB of CSM storage allocated by CSM on its behalf. The vast majority consists of 4 K buffers within the TCP/IP data space.

## 13.6 Performance and tuning

When a situation arises where network performance (often throughput, but sometimes response time) does not meet expectations, the first step is to perform problem source identification, as discussed in this chapter. However, what happens if no “problem” is actually identified? Such a case can happen when the problem is a function of network performance.

Factors that can affect network performance are:

**Maximum transmission unit**

How large can the largest packet be? Is such a packet going to be fragmented on the network?

**Link speed**

How much throughput is available, either as a function of the adapter speed itself or else residual throughput of an adapter that is already carrying other traffic?

**Network latency**

How long does it take data to make the round trip between the endpoints?

Other factors can obviously come into play that are actually outside of the network. For example, how quickly can the endpoint applications respond to packets coming from the network?

**Note:** TCP/IP's dispatching priority (the relative rate at which it is allowed to consume system resources) should be on par with VTAM dispatching priority, which effectively means it should be in the highest service class available. The same goes for critical, time-sensitive, servers like OMPROUTE.

Of course, some network performance problems can be understood to some degree using a simple packet trace. For example, in Example 13-9 on page 322, there are fields such as round trip time (RTT) which provide an accurate estimate of the time it takes a packet to travel out and back along the connection. In addition, the throughput section identifies the basic data throughput characteristics of the connection.

Using a packet trace to analyze network performance, however, is not the right tool for the job. There are many available network performance analysis tools that can reveal details of a network's performance that a packet trace could never identify.

## 13.7 Summary

Because businesses depend heavily on the availability of data processing systems, problems in the network must be addressed quickly. Symptoms of a network problem might include error messages, unusual system behavior, slow response time, or no system response.

The network administrator should first determine the general cause of the problem by reading error messages, checking for system memory dumps, checking to see if software or hardware has changed, and reading the system log. After determining the general cause of the problem, the network

administrator should use the tools and diagnostic aids at hand to determine the specific cause of the problem. Lastly, tuning tasks should be carried out to ensure good network performance.

z/OS has diagnostic aids that the network administrator can use: abend dumps, stand-alone dumps, and supervisor call (SVC) dumps, which the Interactive Problem Control System can format for easier reading. Additionally, VTAM has specific aids, such as First Failure Support Technology, CSDUMPs, network traces, sense codes, VTAM traces, and commands that display the state of VTAM components and resources. TCP/IP has component traces and diagnostic commands (such as the NETSTAT command) that help determine problems in the IP network. Communications Storage Manager (CSM) problems generally manifest themselves as central storage problems. The network administrator can display CSM's use of storage, activate CSM VTAM traces, and dump CSM storage for analysis.

<b>Key terms in this chapter</b>		
abend	authorized problem analysis report (APAR)	Communications Storage Manager (CSM)
component trace	dump	generalized trace facility (GTF)
Netstat	packet trace	program temporary fix (PTF)
supervisor call (SVC)	SYSTCPDA	VTAM internal trace (VIT)

# Notices

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

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

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

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

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

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

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

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

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

## COPYRIGHT LICENSE:

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

# Trademarks

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

Advanced Peer-to-Peer Networking	IBM	S/370
AIX	IMS	S/390®
AS/400	iSeries	System z9
CICS	MQSeries	System/360™
Database 2	MVS	System/370™
DB2	MVS/ESA™	Tivoli
Enterprise Systems Architecture/390®	NetView	VTAM
ESCON	OS/2®	WebSphere
eServer	Parallel Sysplex	z/Architecture
First Failure Support Technology	pSeries	z/OS
FFST	Redbooks	z/VM
FICON	Redbooks (logo)™ 	zSeries
HiperSockets	RACF	z9
	RMF™	
	S/360™	

The following terms are trademarks of other companies:

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

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

Celeron®, Intel Inside®, Intel SpeedStep®, Intel, Itanium®, Pentium®, Xeon®, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

# Glossary

## A

**abend.** abnormal end.

**abend dump.** A kind of dump produced when a program ends abnormally.

**abnormal end.** End of a task, a job, or a subsystem because of an error condition that cannot be resolved by recovery facilities while the task is performed. See also *abnormal termination*.

**abnormal termination.** (1) The cessation of processing prior to planned termination. (2) Abnormal end of task; the termination of a task before its completion because of an error condition that cannot be resolved by recovery facilities while the task is executing. Synonymous with *abend*, *abnormal end*.

**ABR.** area border node.

**access.** The ability to read, update, or otherwise use a resource. Access to protected resources is usually controlled by system software.

**ACK.** acknowledgment field.

**Address Resolution Protocol (ARP).** In the Internet suite of protocols, the protocol that dynamically maps an IP address to an address used by a supporting metropolitan or local area network such as Ethernet or token-ring.

**address space.** A range of contiguous virtual storage addresses that the system creates for the user, batch job, or system task. Unlike a data space, an address space contains user data and programs, as well as system data and programs, some of which are common to all address spaces. Instructions execute in an address space, not a data space. See also *data space*.

**adjacent link station (ALS).** In SNA, a link station directly connected to a given node by a link connection over which network traffic can be carried.

**adjacent node.** (1) In SNA, a node connected to another node by at least one path that connects no other node. (2) In OSI, a node that is attached to the same subnetwork as the local node. An adjacent node can be either a destination node or a relay node.

**adjacent subarea.** A subarea connected by one or more links to another subarea with no intervening subareas. See also *subarea*.

**ADJSSCP.** adjacent SSCP table.

**Advanced Peer-to-Peer Networking (APPN).** An extension to SNA featuring: (a) greater distributed network control that avoids critical hierarchical dependencies, thereby isolating the effects of single points of failure; (b) dynamic exchange of network topology information to foster ease of connection, reconfiguration, and adaptive route selection; (c) dynamic definition of network resources; and (d) automated resource registration and directory lookup. APPN extends the LU 6.2 peer orientation for end-user services to network control and supports multiple LU types, including LU 2, LU 3, and LU 6.2.

**Advanced Peer-to-Peer Networking (APPN) end node.** A node that provides a broad range of end-user services and supports sessions between its local control point (CP) and the CP in an adjacent network node. It uses these sessions to dynamically register its resources with the adjacent CP (its network node server), to send and receive directory search requests, and to obtain management services. An APPN end node can also attach to other end nodes.

**Advanced Peer-to-Peer Networking (APPN) network node.** A node that offers a broad range of end-user services and that can provide the following:

- ▶ Distributed directory services, including registration of its domain resources to a central directory server
- ▶ Topology database exchanges with other APPN network nodes, enabling network nodes throughout the network to select optimal routes for LU-LU sessions based on requested classes of service
- ▶ Session services for its local LUs and client end nodes
- ▶ Intermediate routing services within an APPN network.

**Advanced Program-to-Program Communication (APPC).** An implementation of the SNA LU 6.2 protocol that allows interconnected systems to communicate and share the processing of programs.

**AID.** attention identifier.

**AIW.** APPN Implementation Workshop.

**alert.** To cause the user's terminal to give some audible or visual indication that an error or some other event has occurred.

**alias.** An alternate label; for example, a label and one or more aliases may be used to refer to the same data element or point in a computer program.

**American Standard Code for Information Interchange (ASCII).** A standard code used for information exchange among data processing systems, data communication systems, and associated equipment. ASCII uses a coded character set consisting of 7-bit coded characters. See also *Extended Binary Coded Decimal Interchange Code (EBCDIC)*.

**anonymous user.** A portal user who has not logged in to the portal with a valid user ID and password.

**ANR.** automatic network routing.

**APAR.** authorized program analysis report.

**API.** application programming interface.

**APPC.** advanced program-to-program communication.

**application layer.** In the Open Systems Interconnection (OSI) reference model, the layer that provides means for application processes residing in open systems to exchange information and that contains the application-oriented protocols by which these processes communicate.

**application-instance DVIPA.** A dynamic VIPA activated by an application that explicitly issues a bind() function call to the IP address.

**APPN.** Advanced Peer-to-Peer Networking.

**APPN end node.** See *Advanced Peer-to-Peer Networking (APPN) end node*.

**APPN network.** A collection of interconnected network nodes and their client end nodes.

**APPN network node.** See *Advanced Peer-to-Peer Networking (APPN) network node*.

**APPN node.** An APPN network node or an APPN end node.

**ARB flow.** adaptive rate-based flow.

**area border router (ABR).** Routers that attach to more than one area. All area border routers are part of the backbone, so they must either attach directly to a backbone IP subnet or be connected to another backbone router over a virtual link.

**ARP.** Address Resolution Protocol.

**ARPANET.** A network established by the United States Department of Defense Advanced Research Projects Agency (now the Defense Advanced Research Projects Agency).

**ASCII.** American Standard Code for Information Interchange.

**ASID.** address space identifier.

**Asynchronous Transfer Mode (ATM).** A transfer mode in which the information is organized into cells. It is asynchronous in the sense that the recurrence of cells containing information from an individual user is not necessarily periodic. ATM is specified in international standards such as ATM Forum UNI 3.1.

**ATM.** (1) Asynchronous Transfer Mode. (2) automatic teller machine.

**AT-TLS.** Application Transparent Transport Layer Security.

**authentication.** In computer security, verification of the identity of a user or process and the construction of a data structure that contains the privileges that were granted to the user or process.

**authorization.** The process of granting a user either complete or restricted access to an object, resource, or function.

**automation.** A program or facility that performs certain operations automatically in response to system events or user preferences.

**availability.** The degree to which a system or resource is ready when needed to process data; the percentage of time a system, network, or component can be utilized, within a certain time frame. Generally, the percentage is derived by dividing actual availability time by scheduled availability time. Scheduled outages (no service periods) are not counted against the availability measurement. A service may be unavailable even though the components used to provide the service are all available, and vice-versa.

## B

**backbone.** (1) A set of nodes and their interconnecting links providing the primary data path across a network. (2) In a local area network multiple-bridge ring configuration, a high-speed link to which the rings are connected by means of bridges or routers. A backbone may be configured as a bus or as a ring. (3) In a wide area network, a high-speed link to which nodes or data switching exchanges (DSEs) are connected.

**backup.** Pertaining to a system, device, file, or facility that can be used in the event of a malfunction or loss of data.

**backup host.** A host that is designated as a backup in the event that the distributing host should malfunction. The backup host takes over the IP address of the distributing host when required. See also *distributing host*.

**bandwidth.** (1) The difference, expressed in hertz, between the highest and the lowest frequencies of a range of frequencies. (2) In asynchronous transfer mode (ATM), the capacity of a virtual channel, expressed in terms of peak cell rate (PCR), sustainable cell rate (SCR), and maximum burst size (MBS).

**basic information unit (BIU).** In SNA, the unit of data and control information passed between half-sessions. It consists of a request/response header (RH) followed by a request/response unit (RU).

**batch processing.** A method of running a program or a series of programs in which one or more records (a batch) are processed with little or no action from the user or operator.

**BCP.** base control program.

**BEX.** branch extender.

**bind.** (1) A connection between systems or logical units. (2) In SNA, a request to activate a session between two logical units (LUs). See also *session activation request*.

**border node.** An APPN network node that interconnects APPN networks having independent topology databases in order to support LU-LU sessions between these networks. See *extended border node* and *peripheral border node*.

**boundary function (BF).** In SNA, a capability of a subarea node to provide protocol support for attached peripheral nodes, such as: (a) interconnecting subarea path control and peripheral path control elements, (b) performing session sequence numbering for low-function peripheral nodes, and (c) providing session-level pacing support.

**branch extender.** An extension to the APPN network architecture that appears as a network node to the downstream end nodes in low entry networks and as an end node to the wide area network

**branch network node (BrNN).** See *branch extender*.

**BrEx.** branch extender.

**bridge.** A functional unit that interconnects multiple LANs (locally or remotely) that use the same logical link control protocol, but that can use different medium access control protocols. A bridge forwards a frame to another bridge based on the medium access control (MAC) address.

**BrNN.** branch network node.

**broadcast.** (1) Transmission of the same data to all destinations. (2) Simultaneous transmission of data to more than one destination. (3) Contrast with *multicast*.

**broadcast search.** The propagation of a search request, when the location of a resource is unknown to the requester, to all network nodes in an APPN network. Contrast with *directed Locate search*.

**buffer.** An area of storage that compensates for the different speeds of data flow or timings of events by temporarily holding a block of data that is waiting to be processed or written to an I/O device.

**buffer pool.** An area of memory into which data pages are read, modified, and held during processing.

**bus.** A facility for transferring data between several devices located between two end points, with only one device being able to transmit at a given moment.

**BX.** branch extender.

**byte multiplexer channel.** A multiplexer channel that interleaves bytes of data.

## C

**CA.** certificate authority.

**cage.** See I/O cage.

**capacity.** A measure of how much volume can be handled by a specific resource.

**Carrier Sense Multiple Access with Collision Detection (CSMA/CD).** A media access method that monitors another station's transmissions. If the data station detects another signal during transmission, it stops transmitting, sends a jam signal, then waits for a variable time before trying again.

**CCL.** Communications Controller for Linux.

**CCW.** channel command word.

**CDLC.** Channel Data Link Control protocol.

**CDRM.** cross-domain resource manager.

**CDRSC.** cross-domain resource.

**CDS.** central directory server.

**central directory server.** A network node that provides a repository for information on network resource locations; it also reduces the number of network searches by providing a focal point for queries and broadcast searches, and by caching the results of network searches to avoid later broadcasts for the same information.

**CF.** coupling facility.

**channel.** In mainframe computing, the part of a channel subsystem that manages a single I/O interface between a channel subsystem and a set of control units.

**channel path.** In mainframe computing, the interconnection between a channel and its associated control units.

**channel subsystem (CSS).** A collection of subchannels that directs the flow of information between I/O devices and main storage, relieves the processor of communication tasks, and performs path management functions.

**CHPID.** channel path identifier.

**CICS.** Customer Information Control System.

**CIP.** channel interface processor.

**circuit switching.** (1) A process that, on demand, connects two or more data terminal equipment (DTEs) and permits the exclusive use of a data circuit between them until the connection is released. Synonymous with *line switching*. See also *packet switching*.

**CLAW.** Common Link Access to Workstation.

**client.** A system or process that is dependent on another system or process (usually called the server) to provide it with access to data, services, programs, or resources. See also *server*.

**cluster.** A group of interconnected computers that are working together as one unit.

**CMT.** CHPID mapping tool.

**CNN.** composite network node.

**collision.** An unwanted condition that results from concurrent transmissions on a channel, causing the transmissions to be unintelligible.

**communication.** The process of sending or receiving data between two points of a network.

**communication controller.** A type of communication control unit whose operations are controlled by one or more programs stored and executed in the unit. It manages the details of line control and the routing of data through a network.

**communication line.** Deprecated term for *telecommunication line*.

**Communications Controller for Linux (CCL).** A software product for mainframe servers that emulates the IBM 3745 Communication Controller hardware.

**Communications Server.** IBM software that supports (a) the development and use of application programs among two or more connected systems or workstations, (b) multiple concurrent connections that use a wide range of protocols, and (c) several application programming interfaces (APIs) that may be called concurrently and that are designed for client/server and distributed application programs. Communications Server includes the necessary interfaces for network management and is available on several operating systems (such as AIX, OS/2 Warp, z/OS, and Windows NT). z/OS Communications Server is not available as a stand-alone product. Rather, it is an element of the z/OS operating system. z/OS Communications Server includes the function of these former IBM products: TCP/IP for MVS/ESA and VTAM for MVS/ESA. The VTAM for MVS/ESA function is called Communications Server - SNA Services, and the TCP/IP for MVS/ESA function is called Communications Server - IP Services.

**Communications Storage Manager (CSM).** In z/OS Communications Server, a buffer management technology that reduces performance overhead resulting from the movement of large amounts of data. CSM enables authorized host application programs to put data in buffers that can be addressed and accessed by other authorized host application programs without any need to copy the data.

**component trace.** A service that provides a way for z/OS components to collect problem data about events.

**composite network node.** A type 5 node and its subordinate type 4 nodes that support APPN network node protocols and appear to an attached APPN or LEN node as a single network node.

**concentrator.** (1) In data transmission, a functional unit that permits a common transmission medium to serve more data sources than there are channels currently available within the transmission medium. (2) Any device that combines incoming messages into a single message (concentration) or extracts individual messages from the data sent in a single transmission sequence (deconcentration).

**connection.** In data communications, an association established between functional units for conveying information.

**connectionless protocol.** A transport protocol, such as UDP, that does not require a connection to be established prior to data transfer.

**connection-oriented protocol.** A protocol requiring establishment of a session prior to data transfer.

**connectivity.** (1) The capability of a system or device to be attached to other systems or devices without modification. (2) The capability to attach a variety of functional units without modifying them.

**console.** An input/output device on a computer, reserved for communication between the computer operator or maintenance engineer and the computer.

**control unit.** A device that coordinates and controls the operation of one or more input/output devices, and synchronizes the operation of such devices with the operation of the system as a whole. Synonymous with *controller*.

**controller.** See *control unit*.

**convergence.** The recognition of changes in a network by a dynamic routing protocol.

**CoS.** class of service.

**coupling facility.** A special logical partition that provides high-speed caching, list processing, and locking functions in a sysplex.

**CP.** control point.

**CPC.** central processor complex.

**CP-CP session.** A parallel session between two control points, using LU 6.2 protocols and a mode name of CPSVCMG, on which network services requests and replies are exchanged. Each CP of a given pair has one contention-winner session and one contention-loser session with the other.

**cross-domain resource manager (CDRM).** In VTAM, the function in the system services control point (SSCP) that controls initiation and termination of cross-domain sessions.

**cross-system Coupling Facility (XCF).** Software that allows programs to communicate channel-to-channel, peer-to-peer, across a sysplex. XCF supports program communication and sends program status and signal information between z/OS system images in a sysplex.

**CSA.** common service area.

**CSCL.** carrier-supplied communication line.

**CSM.** Communications Storage Manager.

**CSMA/CD.** carrier sense multiple access with collision detection.

**CSS.** channel subsystem.

**CTC.** channel-to-channel.

**CTRACE.** component trace.

**CU.** control unit.

## D

**daemon.** A program that runs unattended to perform continuous or periodic functions, such as network control.

**DAF.** destination address field.

**DASD.** direct access storage device.

**DB2.** Database 2.

**data integrity.** The condition that exists as long as accidental or intentional destruction, alteration, or loss of data does not occur.

**data link control (DLC).** A set of rules used by nodes on a data link (such as an SDLC link or a token ring) to accomplish an orderly exchange of information.

**data link layer.** In the Open Systems Interconnection reference model, the layer that provides services to transfer data between entities in the network layer over a communication link. The data link layer detects, and possibly corrects, errors that may occur in the physical layer.

**data link switching (DLSw).** A method of transporting network protocols that use IEEE 802.2 logical link control (LLC) type 2. SNA and NetBIOS are examples of protocols that use LLC type 2. See also *encapsulation* and *spoofing*.

**data set.** The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

**data space.** MVS shared memory, parallel to shared memory regions in POSIX. A data space contains data only, which can be shared by multiple address spaces (users) without inadvertently being modified.

**data stream.** (1) A continuous stream of data elements being transmitted, or intended for transmission, in character or binary-digit form, using a defined format. (2) All information (data and control commands) sent over a data link usually in a single read or write operation. For example, a data stream is used to send displays and to receive displays from a workstation device.

**datagram.** In packet switching, a self-contained packet, independent of other packets, which carries information sufficient for routing from the originating data terminal equipment (DTE) to the destination DTE without relying on earlier exchanges between the DTEs and the network.

**DC.** data communication.

**DCB.** device control block.

**DD.** data definition.

**DD card.** Deprecated term for DD statement.

**DD statement.** A job control statement describing a data set associated with a specific job step.

**default.** Pertaining to an attribute, value, or option that is assumed when none is explicitly specified.

**dependent LU.** See *SSCP-dependent LU*.

**dependent LU requester (DLUR).** An APPN end node or network node that: (a) owns dependent LUs in its local node or in adjacently attached nodes; and (b) obtains SSCP services for these dependent LUs from a dependent LU server (DLUS) located elsewhere in an APPN network. The flows of SSCP services between DLUR and DLUS are encapsulated in APPN formats and carried over a special pair of LU 6.2 sessions (referred to as a CP-SVR pipe).

**dependent LU server (DLUS).** An APPN network node that provides SSCP services for dependent LUs owned by dependent LU requesters (DLURs) located elsewhere in an APPN network.

**destination.** (1) Any point or location, such as a node, station, or a particular terminal, to which information is to be sent. (2) An external logical unit (LU) or application program to which messages or other data are directed.

**device control block (DCB).** A control block used by access method routines in storing and retrieving data.

**digital certificate.** An electronic document used to identify an individual, server, company, or some other entity, and to associate a public key with the entity. A digital certificate is issued by a certification authority and is digitally signed by that authority.

**Dijkstra's shortest path algorithm.** An algorithm named after E.W. Dijkstra that finds the shortest path from a source to a destination.

**direct memory access (DMA).** The system facility that allows a device to get direct access to the system or bus memory without the intervention of the system processor.

**directed Locate search.** A search request sent to a specific destination node known to contain a resource, such as a logical unit, to verify the continued presence of the resource at the destination node and to obtain the node's connectivity information for route calculation. Contrast with broadcast search. Synonymous with *directed search*.

**disabled wait state.** Pertaining to a state of a processing unit that prevents the occurrence of certain types of interruptions.

**distributing host.** The designated contact (point of entry) for a sysplex. The distributing host is the normal owner of the IP address that clients out in the network use to connect to the sysplex.

**DLSw.** data link switching.

**DLU.** destination logical unit.

**DLUR.** dependent LU requester.

**DLUS.** dependent LU server.

**DMA.** direct memory access.

**DNS.** domain name server.

**domain.** (1) In SNA communications, the network resources under control of a particular system services control point (SSCP). (2) A part of a network that is administered as a unit with a common protocol.

**domain name server.** In the Internet suite of protocols, a server program that supplies name-to-address translation by mapping domain names to IP addresses. Synonymous with *name server*.

**DS DB.** directory services database.

**dump.** (1) To copy the contents of all or part of storage for the purpose of collecting error information. (2) Data that has been dumped. (3) Synonymous with *memory dump*.

**duplex.** Pertaining to communication in which data can be sent and received at the same time. Synonymous with *full-duplex*. Contrast with *half-duplex*.

**DVIPA.** Dynamic Virtual IP Address.

**dynamic routing protocol.** A protocol that adjusts automatically to network topology or traffic changes.

**dynamic VIPA (DVIPA).** A function that allows the system to move IP addresses in event of an application, TCP/IP stack, or LPAR failure.

**dynamic VPN.** A type of virtual private network that requires a separate server to support the exchange of the keys that are used to encrypt data at each end point.

**dynamic XCF links.** Links using the cross-system coupling facility that can be automatically generated any time TCP/IP becomes active within a sysplex.

## E

**EBCDIC.** Extended Binary Coded Decimal Interchange Code.

**EBN.** extended border node.

**EE.** Enterprise Extender.

**EN.** end node.

**encapsulation.** (1) In communications, a technique used by layered protocols by which a layer adds control information to the protocol data unit (PDU) from the layer it supports. In this respect, the layer encapsulates the data from the supported layer. In the Internet suite of protocols, for example, a packet would contain control information from the physical layer, followed by control information from the network layer, followed by the application protocol data. See also *data link switching (DLSw)*.

**end node.** See *Advanced Peer-to-Peer Networking (APPN) end node*.

**Enterprise Extender.** An extension of SNA High Performance Routing that provides encapsulation of SNA application traffic within UDP frames.

**Enterprise System Connectivity (ESCON).** A peripheral interface for an Enterprise Systems Architecture/390 and mainframe computer. The I/O interface uses ESA/390 logical protocols over a serial interface that configures attached units to a communication fabric.

**ephemeral port number.** In some TCP/IP implementations, a temporary port number assigned to a process for the duration of a call. Ephemeral port numbers are typically assigned to client processes that must provide servers with a client port number so that the server can respond to the correct process.

**ER.** explicit route.

**ESCON.** Enterprise System Connectivity.

**ESCON channel.** A mainframe channel that supports ESCON protocols.

**Ethernet.** A packet-based networking technology for local area networks (LANs) that allows multiple access and handles contention by using Carrier Sense Multiple Access with Collision Detection (CSMA/CD) as the access method. Ethernet is standardized in the IEEE 802.3 specification.

**explicit route.** In SNA, a series of one or more transmission groups that connect two subarea nodes. An explicit route is identified by an origin subarea address, a destination subarea address, an explicit route number, and a reverse explicit route number. Contrast with *virtual route (VR)*.

**Extended Binary Coded Decimal Interchange Code (EBCDIC).** A group of coded character sets that consists of eight-bit coded characters. EBCDIC coded character sets map specified graphic and control characters onto code points, each consisting of 8 bits. EBCDIC is an extension of BCD (Binary-Coded Decimal), which uses only 7 bits for each character.

**extended border node.** A border node that interconnects: (a) APPN networks having different network identifiers; or (b) separate partitions of the same APPN network, where the partitioning is to allow isolated topology subnetworks (or clusters). An extended border node supports intermediate network routing, allowing it to support LU-LU sessions that do not terminate in its native network. Contrast with *peripheral border node*.

## F

**Fast Ethernet.** An Ethernet standard that provides a data rate of 100 Mbps.

**FDDI.** Fiber Distributed Data Interface.

**FEP.** front-end processor.

**FFST.** First Failure Support Technology.

**FICON.** fibre channel connection.

**FID.** format identifier.

**File Transfer Protocol (FTP).** In TCP/IP, an application protocol used for transferring files to and from host computers.

**FIN.** no more data from sender.

**firewall.** A network configuration, usually both hardware and software, that prevents unauthorized traffic into and out of a secure network.

### **First Failure Support Technology (FFST)**

**dump.** A dump produced by First Failure Support Technology, a licensed program that captures information about a potential problem when it occurs.

**format identification field (FID, FID field).** In SNA, a field in each transmission header (TH) that indicates the format of the TH; that is, the presence or absence of certain fields. TH formats differ in accordance with the types of nodes between which they pass.

**forward explicit route.** Explicit routes originating in the host.

**forwarding.** The act of moving a datagram between two different networks or subnetworks.

**frame.** The block of information transmitted between two or more stations in the data link layer of a network. It includes delimiters, control characters, information, and checking characters.

**frame relay.** A protocol for routing frames through the network based on the address field (data link connection identifier) in the frame and for managing the route or virtual connection.

**FRSN.** flow reduction sequence number.

**FTP.** File Transfer Protocol.

**FTPD.** FTP daemon.

**full screen mode.** A form of screen presentation in which the contents of an entire terminal screen can be displayed at once. Full-screen mode is often used for fill-in-the-blanks prompting.

**full-duplex.** See *duplex*.

## G

**gateway.** A device or program used to connect networks or systems with different network architectures. The systems may have different characteristics, such as different communication protocols, different network architecture, or different security policies, in which case the gateway performs a translation role as well as a connection role.

**Gb.** gigabit.

**GB.** gigabyte.

**Gbps.** gigabits per second.

**generalized trace facility (GTF).** In a z/OS environment, a service program that records significant system events, such as supervisor calls and start I/O operations, for the purpose of problem determination.

**Gigabit Ethernet.** A variation of the Ethernet protocol that is capable of transmitting data at one billion bits per second.

**gratuitous ARP.** An unsolicited ARP response.

**GTF.** generalized trace facility.

**guest.** An operating system, such as Linux or z/OS, running in a virtual machine managed by the z/VM Control Program (CP).

**GUI.** graphical user interface.

## H

**half-duplex.** In data communication, pertaining to transmission in only one direction at a time. Contrast with *duplex*.

**handshake.** In Transport Layer Security (TLS), the initial setup of a TLS connection.

**HCD.** hardware configuration definition.

**hardware configuration definition (HCD).** An interactive interface in z/OS that is used to define hardware configurations to the operating system and the channel subsystem.

**Hardware Management Console (HMC).** A console used to monitor and control hardware, such as the IBM System z9 processors.

**HFS.** hierarchical file system.

**HiperSockets.** A technology that provides high-speed TCP/IP connectivity within a central processor complex. It eliminates the need for any physical cabling or external networking connection between servers running in different LPARs.

**HMC.** Hardware Management Console.

**hop.** (1) In APPN, a portion of a route that has no intermediate nodes. It consists of only a single transmission group connecting adjacent nodes. (2) To the routing layer, the logical distance between two nodes in a network.

**host.** A computer that is connected to a network and provides an access point to that network. The host can be a client, a server, or both a client and server simultaneously.

**HPR.** high performance routing.

**HTTP.** Hypertext Transfer Protocol.

**HTTP server.** A program that enables a computer that uses the Hypertext Transfer Protocol (HTTP) to serve objects by responding to requests from other programs, such as browsers.

**hub.** In a network, a point at which circuits are either connected or switched. For example, in a star network, the hub is the central node; in a star/ring network, it is the location of wiring concentrators.

## I

**I/O cage.** A physical area of the processor frame where connections to the central processor complex are made.

**IAC.** Interpret As Command.

**IBM.** International Business Machines Corporation.

**IBM Security Server.** An IBM licensed program that provides access control by identifying users to the system; verifying users of the system; authorizing access to protected resources; logging detected, unauthorized attempts to enter the system; and logging detected accesses to protected resources. Also known as Resource Access Control Facility (RACF).

**ICMP.** Internet Control Message Protocol.

**ICN.** interchange network node.

**IDS.** intrusion detection services.

**IEEE.** Institute of Electrical and Electronic Engineers.

**IETF.** Internet Engineering Task Force.

**IKE.** Internet Key Exchange.

**IMS.** Information Management System.

**independent LU.** See *SSCP-independent LU*.

**Integrated Services Digital Network (ISDN).** An international communications standard for sending voice, video, and data over digital telephone lines.

**interchange node.** (1) A node that acts as both an APPN network node and a type 5 subarea node to transform APPN protocols to subarea protocols and vice versa. (2) Contrast with *migration data host*.

**Internet.** The worldwide collection of interconnected networks that use the Internet suite of protocols and permit public access.

**Internet Control Message Protocol (ICMP).** An Internet protocol that is used by a gateway to communicate with a source host, for example, to report an error in a datagram.

**Internet Protocol (IP).** A protocol that routes data through a network or interconnected networks. Internet Protocol (IP) acts as an intermediary between the higher protocol layers and the physical network.

**intranet.** A private network that integrates Internet standards and applications (such as Web browsers) with an organization's existing computer networking infrastructure.

**intrusion detection service.** Software that detects attempts or successful attacks on monitored resources that are part of a network or host system.

**IOCDs.** I/O configuration data set.

**IOCP.** I/O configuration program.

**IODF.** I/O definition file.

**IOP.** I/O program.

**IP.** Internet Protocol.

**IP address.** The unique 32-bit address (or, for IP version 6, the 128-bit address) that specifies the location of each device or workstation in the Internet. For example, 9.67.97.103 is an IP address. The address field contains two parts: the first part is the network address; the second part is the host number.

**IP layer.** Synonym for *network layer*.

**IP network.** A network that consists of subnetworks that are connected through the Internet Protocol.

**IP route.** A network path between any two IP addressable points in a network.

**IP Security Architecture.** A collection of Internet Engineering Task Force (IETF) standards that define an architecture at the Internet Protocol (IP) layer to protect IP traffic by using various security services.

**IPA.** IP network availability.

**IPCS.** Interactive Problem Control System.

**IPL.** initial program load.

**IPSec.** IP Security.

**IPv4.** Internet Protocol version 4.

**IPv6.** Internet Protocol version 6.

**IQDIO.** internal Queued Direct I/O.

**ISC.** inter-system coupling.

**ISDN.** Integrated Services Digital Network.

**ISPF.** Interactive System Productivity Facility.

**ITSO.** International Technical Support Organization.

## J

**JCL.** job control language.

**JES.** job entry subsystem.

**job.** A unit of work for an operating system. Jobs are defined by JCL statements.

## K

**Kb.** kilobit.

**KB.** kilobyte.

**Kbps.** Kilobits per second.

**key ring.** In computer security, a file that contains public keys, private keys, trusted roots, and certificates.

## L

**LAN.** local area network.

**LAN segment.** (1) Any portion of a LAN (for example, a bus or ring) that can operate independently, but that is connected to other parts of the network by means of bridges. (2) A ring or bus network without bridges.

**LCSS.** logical channel subsystem.

**LDAP.** Lightweight Directory Access Protocol.

**LEN node.** low-entry network (LEN) node.

**line switching.** Synonym for *circuit switching*.

**link.** In data communications, a transmission medium and data link control component that together transmit data between adjacent nodes.

**link layer.** See *data link layer*.

**link state.** In routing protocols, the advertised information about the usable interfaces and reachable neighbors of a router or network. The protocol's topological database is formed from the collected link-state advertisements.

**link station.** In SNA, a named resource within an APPN or a subarea node that represents the connection to another APPN or subarea node that is attached by an APPN or a subarea link. In the resource hierarchy in a subarea network, the link station is subordinate to the subarea link.

**LLC.** logical link control.

**load balancing.** The monitoring and management of the workload on servers. If one server exceeds its workload, requests are forwarded to another server with more capacity.

**local area network (LAN).** A network that connects several devices in a limited area (such as a single building or campus) and that can be connected to a larger network.

**local topology database.** A database in an APPN or LEN node containing an entry for each transmission group (TG) having at least one end node for an endpoint. In an end node, the database has one entry for each TG connecting to the node. In a network node, the database has an entry for each TG connecting the network node to an end node. Each entry describes the current characteristics of the TG that it represents. A network node has both a local and a network topology database, while an end node has only a local topology database.

**Locate search.** The means which directory services in a node uses to find a resource that is not in that node. The Locate search enables directory services to ask the directory services components in other APPN nodes for information on the target resource. See also *broadcast search* and *directed Locate search*.

**logical channel subsystem (LCSS).** A channel subsystem structure that provides channel path and subchannel controls for configuring from one to four channel subsystem images. Each channel subsystem image can be configured with up to 256 channel paths, and each logical partition has access to one channel subsystem image.

**logical partition (LPAR).** A subset of a single system that contains resources (processors, memory, and input/output devices). A logical partition operates as an independent system. If hardware requirements are met, multiple logical partitions can exist within a system.

**logical unit (LU).** An access point through which a user or application program accesses the SNA network to communicate with another user or application program. An LU can support at least two sessions—one with an SSCP and one with another LU—and may be capable of supporting many sessions with other LUs.

**LOGREC.** log recording data set.

**low entry networking (LEN) node.** A node that provides a range of end-user services, attaches directly to other nodes using peer protocols, and derives network services implicitly from an adjacent APPN network node, that is, without the direct use of CP-CP sessions.

**LPAR.** logical partition.

**LSPR.** Large System Performance Reference.

**LU-LU session.** A logical connection between two logical units (LUs) in an SNA network that typically provides communication between two users.

## M

**MAC.** media access control (MAC).

**MAC address.** A standardized data link layer address required for every port or device that connects to a local-area network (LAN). Other devices in the network use these addresses to locate specific ports in the network and to create and update routing tables and data structures. MAC addresses are 6 bytes long and are controlled by the IEEE.

**macro instruction.** An instruction in a source language that is to be replaced by a defined sequence of instructions in the same source language, and that may also specify values for parameters in the replaced instructions.

**mainframe.** A computer, usually in a computer center, with extensive capabilities and resources to which other computers may be connected so that they can share facilities.

**major node.** In VTAM, a set of resources that can be activated and deactivated as a group. See also *minor node*.

**Mb.** megabit.

**MB.** megabyte.

**Mbps.** megabits per second.

**MDH.** migration data host.

**media access control (MAC).** In LANs, the sublayer of the data link control layer that supports media-dependent functions and uses the services of the physical layer to provide services to the logical link control (LLC) sublayer. The MAC sublayer includes the method of determining when a device has access to the transmission medium.

**memory dump.** See *dump*.

**metropolitan area network (MAN).** (1) A network formed by the interconnection of two or more networks which may operate at higher speed than those networks, may cross administrative boundaries, and may use multiple access methods. Contrast with *local area network (LAN)* and *wide area network (WAN)*.

**MIF.** multiple image facility.

**migration data host.** A node that acts as both an APPN end node and a type 5 subarea node. Contrast with *interchange node*.

**minor node.** In VTAM, a uniquely defined resource within a major node.

**MLTG.** multilink transmission group.

**MNLB.** multi-node load balancing.

**modem (modulator-demodulator).** A device that converts digital data from a computer to an analog signal that can be transmitted on a telecommunication line, and converts the analog signal received to data for the computer.

**Monitor I, II, III.** Components of the Resource Measurement Facility (RMF).

**MPC.** multipath channel.

**MTU.** maximum transmission unit.

**multicast.** (1) Transmission of the same data to a selected group of destinations. (2) A special form of broadcast in which copies of a packet are delivered to only a subset of all possible destinations. (3) Contrast with *broadcast*.

**multiple image facility (MIF).** A facility that allows multiple logical partitions to share ESCON channels (and FICON channels) and optionally to share any of the control units and associated I/O devices configured to these shared channels. The sharing can reduce channel requirements, improve channel utilization, and improve I/O connectivity.

**multiplexer channel.** A channel designed to operate with a number of I/O devices simultaneously. Several I/O devices can transfer records at the same time by interleaving items of data.

**MVS.** Multiple Virtual Storage, an earlier form of the z/OS operating system. In current usage, the term MVS refers to those services and functions of z/OS other than z/OS UNIX system services, such as those provided by the base control program (BCP), a base element of z/OS.

## N

**name server.** In the Internet suite of protocols, synonym for *domain name server*.

**NAU.** Network accessible unit.

**NCP.** Network Control Program.

**network.** In data communications, a configuration in which two or more locations are physically connected for the purpose of exchanging data.

**network accessible unit (NAU).** A logical unit (LU), physical unit (PU), control point (CP), or system services control point (SSCP). It is the origin or the destination of information transmitted by the path control network. Synonymous with *network addressable unit*.

**network administrator.** A person who manages the use and maintenance of a network.

**network controller.** A concentrator and protocol converter used with SDLC links. By converting protocols, which manage the way data is sent and received, the IBM 3710 Network Controller allows the use of non-SNA devices with an SNA host processor.

**network hardware technician.** A person with specific skills and tools for supporting the physical network.

**network ID.** (1) In TCP/IP, that part of the IP address that defines a network. (2) A 1- to 8-byte customer-selected name or an 8-byte IBM-registered name that uniquely identifies a specific subnetwork.

**network interface card (NIC).** A printed circuit board that plugs into a personal computer, server, or workstation. It controls the exchange of data over a network and provides the electronic functions for the data link protocol or access method, such as token ring or Ethernet.

**network layer.** In Open Systems Interconnection (OSI) architecture, the layer that is responsible for routing, switching, and link-layer access across the OSI environment.

**network node (NN).** See *Advanced Peer-to-Peer Networking (APPN) network node*.

**network node server.** An APPN network node that provides network services for its local LUs and client end nodes.

**network operator.** (1) A person who controls the operation of all or part of a network. (2) In a multiple-domain network, a person or program responsible for controlling all domains.

**network protocol.** A communications protocol from the network layer of the Open Systems Interconnect (OSI) network architecture, such as the Internet Protocol (IP).

**network topology database.** The representation of the current connectivity between the network nodes within an APPN network. It includes: (a) entries for all network nodes and the transmission groups interconnecting them; and (b) entries for all virtual routing nodes to which network nodes are attached.

**NFS.** network file system.

**NFTP.** NetView File Transfer.

**NHDR.** network layer header.

**NIC.** network interface card.

**NLP.** network layer packet.

**NN.** network node.

**NPM.** NetView Performance Monitor.

**O**

**OAF.** origin address field.

**OLTP.** online transaction processing.

**OLU.** originating logical unit.

**OMPROUTE server.** The routing daemon on z/OS capable of handling both OSPF and RIP interfaces concurrently.

**Open Shortest Path First (OSPF).** In the Internet suite of protocols, a function that provides intradomain information transfer. An alternative to the Routing Information Protocol (RIP), OSPF allows the lowest-cost routing and handles routing in large regional or corporate networks.

**Open Systems Adapter.** An integrated IBM mainframe hardware feature that combines the functions of an I/O channel with the functions of a network port to provide direct connectivity between mainframe applications and their clients on the attached network.

**Open Systems Interconnection (OSI).** The interconnection of open systems in accordance with standards of the International Organization for Standardization (ISO) for the exchange of information.

**OpenSSH.** Open Secure Shell.

**ORB.** operation request block.

**OSA.** Open Systems Adapter.

**OSI.** Open Systems Interconnection.

**OSN.** OSA for NCP.

**OSPF.** Open Shortest Path First.

## P

**packet.** In data communication, a sequence of binary digits, including data and control signals, that is transmitted and switched as a composite whole. See also *frame*.

**packet mode operation.** See *packet switching*.

**packet switching.** (1) The process of routing and transferring data by means of addressed packets so that a channel is occupied only during transmission of a packet. On completion of the transmission, the channel is made available for transfer of other packets. (2) Synonymous with *packet mode operation*. See also *circuit switching*.

**parallel channel.** A channel having a System/360 and System/370 channel-to-control-unit I/O interface that uses bus-and-tag cables as a transmission medium. Contrast with *ESCON channel*.

**parallel OEMI.** A reference to an older IBM standard for a computer peripheral interface, which defines the IBM S/360 and S/370 channel to control unit interface. This interface uses ESA/390 logical protocols over a common bus that configures attached units in a multi-drop bus topology.

**Parallel Sysplex.** A sysplex that uses one or more coupling facilities.

**parameter (parm).** A value or reference passed to a function, command, or program that serves as input or controls actions. The value is supplied by a user or by another program or process.

**parmlib.** All the members in the SYS1.PARMLIB partitioned data set that contain parameters setting the limits and controlling the behavior of z/OS.

**partitioned data set (PDS).** A data set in direct-access storage that is divided into partitions, which are called members. Each partition can contain a program, part of a program, or data. The term partitioned data set is synonymous with program library.

**path information unit (PIU).** A message unit consisting of a transmission header (TH) alone, or a TH followed by a basic information unit (BIU) or a BIU segment.

**PCHID.** physical channel identifier.

**PCOM.** IBM Personal Communications.

**peer network.** A network in which every resource is self-contained and controls its own resources.

**peripheral border node.** A border node that interconnects adjacent APPN networks having different network identifiers in order to support LU-LU sessions that have one partner LU in its native network. Contrast with *extended border node*.

**peripheral device.** Any device that can communicate with a particular computer; for example, input/output units, auxiliary storage.

**peripheral node.** A node that uses local addresses for routing and therefore is not affected by changes in network addresses. A peripheral node requires boundary-function assistance from an adjacent subarea node. A peripheral node can be a type 1, 2.0, or 2.1 node connected to a subarea boundary node.

**peripheral subnetwork boundary.** A connection over a subnetwork boundary between a border and a network node with no border node function.

**physical channel ID (PCHID).** A number assigned by the machine to a physical channel location.

**physical layer.** In the Open Systems Interconnection reference model, the layer that provides the mechanical, electrical, functional, and procedural means to establish, maintain, and release physical connections over the transmission medium.

**physical unit (PU).** In SNA, one of three types of network addressable units. A physical unit exists in each node of an SNA network to manage and monitor the resources (such as attached links and adjacent link stations) of a node, as requested by a system services control point logical unit (SSCP-LU) session.

**PIN.** personal identification number.

**PIU.** path information unit.

**PLU.** primary logical unit.

**point of presence.** A system that has been identified as a contact point for another subnetwork for the purposes of collecting topology information.

**point-to-multipoint network.** A network in which there are many hosts directly attached within the scope of a single network ID.

**point-to-point network.** Pertaining to data transmission between two locations without the use of any intermediate display station or computer.

**policy.** A set of rules that are applied to managed resources.

**port.** (1) An access point for data entry or exit. (2) A connector on a device to which cables for other devices such as display stations and printers are attached. (3) The representation of a physical connection to the link hardware. A port is sometimes referred to as an adapter; however, there can be more than one port on an adapter. There may be one or more ports controlled by a single data link control process. (4) In the Internet suite of protocols, a 16-bit number used to communicate between TCP or the User Datagram Protocol (UDP) and a higher-level protocol or application. Some protocols, such as File Transfer Protocol (FTP) and Simple Mail Transfer Protocol (SMTP), use the same well-known port number in all TCP/IP implementations. (5) An abstraction used by transport protocols to distinguish among multiple destinations within a host machine.

**port number.** The part of a socket address that identifies a port within a host.

**PoS.** point-of-sale.

**power-on reset.** A key sequence that restarts the operating system (or other program) without turning off the electrical power of the system.

**production system.** A system where application programs that are already developed and tested run on a regular basis.

**program temporary fix (PTF).** A fix that is made available to all customers. A program temporary fix is tested by IBM.

**protocol.** (1) The meaning of, and the sequencing rules for, requests and responses used for managing a network, transferring data, and synchronizing the states of network components.

(2) A set of semantic and syntactic rules that determines the behavior of functional units in achieving communication.

(3) A set of rules controlling the communication and transfer of data between two or more devices or systems in a communications network.

**protocol stack.** A set of network protocol layers and software that work together to process the protocols.

**protocol suite.** A set of protocols that cooperate to handle the transmission tasks for a communication system.

**PSH.** push data function.

**PSS.** process scheduling services.

**PTF.** program temporary fix.

**PU.** physical unit.

**PUT.** program update tape.

## Q

**QDIO.** Queued Direct I/O.

**queued direct I/O (QDIO).** A hardware channel architecture for direct data exchange with I/O devices, where both the I/O device and the program running on the server reference main storage directly through a set of data queues. The QDIO architecture is used by Open Systems Adapter-Express (OSA-Express), HiperSockets, and Fiber Channel Protocol (FCP) channels.

## R

**RACF.** Resource Access Control Facility.

**RAS.** reliability, availability, serviceability.

**recoverability.** The extent to which the system can be restored to an operational condition after a system failure.

**redundancy.** The use of several identical functional units, such as several disk drives or power supply systems, within one computer system in order to provide data security and a certain degree of fault tolerance in case of hardware failures.

**reliability.** A measurement of the ability of a system to continue processing without failure.

**Request For Comment (RFC).** In Internet communications, a document that describes a part of the Internet suite of protocols and related experiments. All Internet standards are documented as RFCs.

**request header (RH).** The control information that precedes a request unit (RU).

**request unit (RU).** A message unit that contains control information, end-user data, or both.

**resolver.** In TCP/IP, a program or subroutine that obtains information from a domain name server or a local table for use by an application program.

**Resource Access Control Facility.** See *IBM Security Server*.

**Resource Management Facility (RMF).** Software that measures and reports on the performance and availability of a system.

**response unit (RU).** A message unit that acknowledges a request unit. It may contain prefix information received in a request unit. If positive, the response unit may contain additional information (such as session parameters in response to BIND SESSION). If negative, the response unit contains sense data defining the exception condition.

**reverse explicit route.** Explicit routes that terminate in the host and must use the same set of subarea nodes and transmission groups as their corresponding forward explicit route.

**RFC.** request for comment.

**RH.** request/response header.

**RIP.** Routing Information Protocol.

**RMF.** Resource Measurement Facility.

**route selection services (RSS).** A subcomponent of the topology and routing services component that determines the preferred route between a specified pair of nodes for a given class of service.

**router.** (1) A computer that determines the path of network traffic flow. The path selection is made from several paths based on information obtained from specific protocols, algorithms that attempt to identify the shortest or best path, and other criteria such as metrics or protocol-specific destination addresses. (2) An attaching device that connects two LAN segments, which use similar or different architectures, at the reference model network layer. (3) In OSI terminology, a function that determines a path by which an entity can be reached. (4) In TCP/IP, synonymous with *gateway*. (5) Contrast with *bridge*.

**routing.** The assignment of the path by which a message is to reach its destination.

**Routing Information Protocol (RIP).** In the Internet suite of protocols, a protocol used to exchange intradomain routing information and to determine optimum routes between internet hosts. This protocol determines optimum routes on the basis of route metrics, not link transmission speed.

**routing protocol.** A technique used by a router to find other routers and to remain up to date about the best way to get to reachable networks.

**routing table.** A collection of routes used to direct datagram forwarding or to establish a connection. The information is passed among routers to identify network topology and destination feasibility.

**RRI.** request/response indicator.

**RSN.** resource sequence number.

**RST.** reset the connection.

**RTP.** rapid transport protocol.

**RTT.** round trip time.

**RU.** request/response unit.

## S

**SAF.** system authorization facility.

**SAP.** service access point.

**SATF.** shared access transport facility.

**scan attack.** An attack in which a host on the network is trying to determine what ports are open on the target host. The host doing the scan may later be the same host that does a more virulent attack.

**SDLC.** synchronous data link control.

**SDSF.** System Display and Search Facility.

**SDUMP macro instruction.** A z/OS macro that can be invoked by authorized programs to take a fast unformatted dump of virtual storage to a data set.

**SE.** Support Element.

**Secure Sockets Layer (SSL).** A security protocol that provides communication privacy. With SSL, client/server applications can communicate in a way that is designed to prevent eavesdropping, tampering, and message forgery.

**sendmail.** The mail server that uses the Simple Mail Transfer Protocol (SMTP) to route mail from one host to another on the network.

**sense code.** A diagnostic code that helps in problem determination.

**server.** (1) A functional unit that provides shared services to workstations over a network; for example, a file server, a print server, a mail server. (2) In a network, a data station that provides facilities to other stations; for example, a file server, a print server, a mail server. (3) In the AIX operating system, an application program that usually runs in the background and is controlled by the system program controller.

**service access point (SAP).** (1) In Open Systems Interconnection (OSI) architecture, the point at which the services of a layer are provided by an entity of that layer to an entity of the next higher layer. (2) A logical point made available by an adapter where information can be received and transmitted. A single service access point can have many links terminating in it. (3) A logical address that allows a system to route data between a remote device and the appropriate communications support. (4) The identification of the services provided by a specific communication service provider to one of its users. For example, the Internet Protocol (IP) uses the services of a token-ring adapter. The service access point, in this case, is the name by which IP knows the adapter that is the token-ring address.

**session.** (1) A logical connection between two network accessible units (NAUs) that can be activated, tailored to provide various protocols, and deactivated, as requested. Each session is uniquely identified in a transmission header (TH) accompanying any transmissions exchanged during the session. (2) A logical or virtual connection between two stations, programs, or devices on a network that allows the two elements to communicate and exchange data, or the activities that occur during the establishment, maintenance, and release of the connection. A session can be activated and deactivated as requested. See also *transaction*.

**session activation request.** In SNA, a request that activates a session between two network accessible units (NAUs) and specifies session parameters that control various protocols during session activity; for example, BIND and ACTPU.

**sftp.** secure FTP.

**shell.** A program that interprets sequences of text input as commands. It may operate on an input stream, or it may interactively prompt and read commands from a terminal.

**SIGA.** signal adapter.

**SLA.** service level agreement.

**SLU.** secondary logical unit.

**SMTP.** Simple Mail Transfer Protocol.

**SNA.** Systems Network Architecture.

**SNA network interconnection (SNI).** The connection, by gateways, of two or more independent SNA networks to allow communication between logical units in those networks. The individual SNA networks retain their independence.

**SNASw.** SNA Switching Services.

**SNI.** SNA network interconnection.

**SNMP.** Simple Network Management Protocol.

**socket.** A unique host identifier created by the concatenation of a port identifier with a TCP/IP address.

**spoofing.** For data links, a technique in which a protocol initiated from an end station is acknowledged and processed by an intermediate node on behalf of the final destination. In IBM 6611 data link switching, for example, SNA frames are encapsulated into TCP/IP packets for transport across a non-SNA wide area network, unpacked by another IBM 6611, and passed to the final destination. A benefit of spoofing is the prevention of end-to-end session timeouts.

**SSCH.** start subchannel.

**SSCP.** system services control point.

**SSCP-dependent LU.** An LU that requires assistance from a system services control point (SSCP) in order to initiate an LU-LU session. It requires an SSCP-LU session.

**SSCP-independent LU.** An LU that is able to activate an LU-LU session (that is, send a BIND request) without assistance from an SSCP. It does not have an SSCP-LU session. Currently, only an LU 6.2 can be an independent LU.

**SSCP-LU session.** In SNA, a session between a system services control point (SSCP) and a logical unit (LU). The session enables the LU to request the SSCP to help initiate LU-LU sessions.

**SSCP-SSCP session.** In SNA, a session between the system services control point (SSCP) in one domain and the SSCP in another domain. An SSCP-SSCP session is used to initiate and terminate cross-domain LU-LU sessions.

**SSH.** Secure Shell.

**SSID.** subsystem identifier.

**SSL.** Secure Socket Layer.

**stand-alone dump.** A kind of dump produced by an operator using a stand-alone dump program, a special program that can run by itself when an operating system is disabled.

**star topology.** In network architecture, a network topology in which every node on the network is connected to a central node or "hub," through which they communicate with each other.

**start option.** In VTAM, a user-specified or IBM-supplied option that determines certain conditions that are to exist during the time a VTAM system is operating. Start options can be predefined or specified when VTAM is started.

**started task.** In MVS, an address space that runs unattended as the result of a START command. Started tasks are generally used for critical applications. The UNIX equivalent is a daemon.

**state-oriented protocol.** A characteristic of the OSPF protocol, in which interfaces and neighboring routers are always classified as being in a particular state. Events on the network causes these states to change in a pre-determined way, providing a predictability and control to the OSPF routers on the network.

**static routing.** A method of setting paths between hosts, networks, or both, by manually entering routes into the routing table. Static routes are not affected by routing daemons and must be updated manually.

**STI.** self-timed interface.

**stub area.** In the OSPF protocol, a routing area for which packets can flow into and out of, but not through.

**subarea.** A portion of the SNA network consisting of a subarea node, attached peripheral nodes, and associated resources. Within a subarea node, all network accessible units (NAUs), links, and adjacent link stations (in attached peripheral or subarea nodes) that are addressable within the subarea share a common subarea address and have distinct element addresses.

**subarea network.** Interconnected subareas, their directly attached peripheral nodes, and the transmission groups that connect them.

**subnet.** Synonym for *subnetwork*.

**subnetwork.** (1) Any group of nodes that have a set of common characteristics, such as the same network ID. (2) A distinct partitioned piece of an internet network represented by two or more sets of addresses that are subsets of the network's range of addresses. (3) Synonymous with *subnet*.

**supervisor call (SVC).** A request that serves as the interface into operating system functions, such as allocating storage. The SVC protects the operating system from inappropriate user entry. All operating system requests must be handled by SVCs.

**support element (SE).** (1) An internal control element of a processor that assists in many of the processor operational functions. (2) A hardware unit that provides communications, monitoring, and diagnostic functions to a central processor complex (CPC).

**SVC.** supervisor call.

**SVC dump.** A memory dump executed by a supervisor call (system macroinstruction).

**switched multimegabit data service (SMDS).** A high-speed technology offered by telephone companies in the United States.

**SYN.** synchronize sequence numbers.

**SYN flood.** A type of denial of service attack in which a series of SYN packets are received in a short period of time. A SYN is the first packet received when a remote host is attempting a TCP connection.

**Synchronous Data Link Control (SDLC).** A protocol for managing synchronous information transfer over a data link connection.

**sysplex distributor.** A software function in z/OS that increases availability through a combination of dynamic VIPA and the z/OS Workload Manager.

**System Authorization Facility (SAF).** An interface defined by z/OS that enables programs to use system authorization services in order to protect access to resources such as data sets and z/OS commands. The IBM Security Server is a product that uses the SAF interface.

**System Display and Search Facility (SDSF).** Part of z/OS that provides detailed information about the jobs and resources in the system, including the job entry subsystem, JES2.

**system operator.** The person responsible for performing system-oriented procedures.

**system services control point (SSCP).** A focal point in an SNA network for managing configuration, coordinating network-operator and problem-determination requests, and providing directory support or other session services for network users. Multiple SSCPs, cooperating as peers, can divide the network into domains of control, with each SSCP controlling the physical and logical units in its domain.

**Systems Network Architecture (SNA).** The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks. The layered structure of SNA allows the ultimate origins and destinations of information, that is, the users, to be independent of and unaffected by the specific SNA network services and facilities used for information exchange.

**systems programmer.** A programmer who plans, generates, maintains, extends, and controls the use of an operating system with the aim of improving overall productivity of an installation.

## T

**T2.0.** type 2.0 node.

**T2.1.** type 2.1 node.

**T4.** type 4 node.

**T5.** type 5 node.

**TCP.** Transmission Control Protocol.

**TDM.** topology database manager.

**TDU.** topology data update.

**telecommunication line.** (1) The portion of a data circuit external to a data circuit-terminating equipment (DCE) that connects the DCE to a data-switching exchange (DSE), that connects a DCE to one or more other DCEs, or that connects a DSE to another DSE. (2) Any physical medium, such as a wire or microwave beam, that is used to transmit data. Synonymous with *transmission line*.

**Telnet.** In the Internet suite of protocols, a protocol that provides remote terminal connection service. It allows users of one host to log on to a remote host and interact as directly attached terminal users of that host.

**terminal.** In data communication, a device, usually equipped with a keyboard and display device, capable of sending and receiving information.

**TG.** transmission group.

**TGV.** transmission group vector.

**TH.** transmission header.

**throughput.** A measure of the amount of information transmitted over a network in a given period of time. Throughput is a measure of performance. It is generally measured in bits per second (bps), kilobits per second (Kbps), megabits per second (Mbps), or gigabits per second (Gbps).

**TLS.** Transport Layer Security.

**TN3270.** Telnet/3270.

**TN3270E.** Telnet/3270 Enhanced.

**token ring network.** (1) According to IEEE 802.5, network technology that controls media access by passing a token (special packet or frame) between media-attached stations. (2) A FDDI or IEEE 802.5 network with a ring topology that passes tokens from one attaching ring station (node) to another. (3) See also *local area network (LAN)*.

**topology database update (TDU).** A message about a new or changed link or node that is broadcast among APPN network nodes to maintain the network topology database, which is fully replicated in each network node. A TDU contains information that identifies the following:

- ▶ The sending node
- ▶ The node and link characteristics of various resources in the network
- ▶ The sequence number of the most recent update for each of the resources described.

**topology database.** See *local topology database* and *network topology database*.

**ToS.** type of service.

**TP.** transmission priority.

**transaction.** A unit of processing consisting of one or more application programs, affecting one or more objects, that is initiated by a single request, often from a terminal.

**transit area.** In the OSPF protocol, a routing area that can have traffic passing through it.

**Transmission Control Protocol (TCP).** A communications protocol used in the Internet and in any network that follows the U.S. Department of Defense standards for internetwork protocol. TCP provides a reliable host-to-host protocol between hosts in packet-switched communications networks and in interconnected systems of such networks. It uses the Internet Protocol (IP) as the underlying protocol.

**transmission group (TG).** (1) A connection between adjacent nodes that is identified by a transmission group number. (2) In a subarea network, a single link or a group of links between adjacent nodes. When a transmission group consists of a group of links, the links are viewed as a single logical link, and the transmission group is called a multilink transmission group (MLTG). A mixed-media multilink transmission group (MMMLTG) is one that contains links of different medium types (for example, token-ring, switched SDLC, nonswitched SDLC, and frame-relay links). (3) In an APPN network, a single link between adjacent nodes.

**transmission header (TH).** Control information, optionally followed by a basic information unit (BIU) or a BIU segment, that is created and used by path control to route message units and to control their flow within the network. See also *path information unit*.

**transmission line.** See *telecommunication line*.

**transmission priority.** A rank assigned to a message unit that determines its precedence for being selected by the path control component in each node along a route for forwarding to the next node in the route.

**transport layer.** A network service that provides end-to-end communications between two parties, while hiding the details of the communications network. The TCP and ISO TP4 transport protocols provide full-duplex virtual circuits on which delivery is reliable, error free, sequenced, and duplicate free. UDP provides no guarantees (the connectionless RPC protocol provides some guarantees on top of UDP).

**Transport Layer Security (TLS).** A protocol standard that uses encryption to provide confidentiality and authentication between two TCP/IP applications.

**transport protocol.** A specification of the rules governing the exchange of information between components of a transport network.

**TRL.** transport resource list.

**TRLE.** transport resource list element.

**trunk.** In telephony, circuits that connect two switching systems, as opposed to connecting a customer line to a switching system.

**TSO/E.** Time Sharing Option/Extensions.

**TSR.** target service responsiveness.

**twisted pair.** A transmission medium that consists of two insulated electrical conductors twisted together to reduce noise.

## U

**UDP.** User Datagram Protocol.

**UID.** user identifier.

**URG.** urgent flag.

**User Datagram Protocol (UDP).** In the Internet suite of protocols, a protocol that provides unreliable, connectionless datagram service. It enables an application program on one machine or process to send a datagram to an application program on another machine or process. UDP uses the Internet Protocol (IP) to deliver datagrams.

**UTP.** unshielded twisted pair.

## V

**VIPA.** virtual IP address.

**virtual IP address (VIPA).** An IP address that is not associated with any physical interface.

**virtual private network (VPN).** A general term to describe a secure tunnel (data stream) between two endpoints. The term does not describe a protocol. The industry standard protocol for a VPN is an architecture called IP Security Architecture (IPSec).

**virtual route (VR).** (1) In SNA, either (a) a logical connection between two subarea nodes that is physically realized as a particular explicit route or (b) a logical connection that is contained wholly within a subarea node for intranode sessions. A virtual route between distinct subarea nodes imposes a transmission priority on the underlying explicit route, provides flow control through virtual route pacing, and provides data integrity through sequence numbering of path information units (PIUs).  
(2) Contrast with *explicit route (ER)*.

**Virtual Telecommunications Access Method (VTAM).** Software that implements the Systems Network Architecture. Now known as Communications Server - SNA Services.

**VIT.** VTAM internal trace.

**VLAN.** virtual LAN.

**VPN.** virtual private network.

**VR.** virtual route.

**VRN.** virtual routing node.

**VTAM.** Virtual Telecommunications Access Method.

## W

**WAN.** wide area network.

**WCC.** write control character.

**well-known port.** In Internet communications, one of a set of preassigned protocol port numbers that address specific functions used by transport level protocols (for example, TCP and UDP).

**wide area network (WAN).** A network that provides communication services between devices in a geographic area larger than that served by a local area network (LAN) or a metropolitan area network (MAN).

**Wi-Fi.** wireless fidelity.

**WLM.** Workload Manager.

**Workload Manager (WLM).** A z/OS component that prioritizes workloads running on z/OS and matches workloads with available resources.

**WSF.** write structured field.

## X

**XCA.** external communication adapter.

**XCF.** cross-system coupling facility.

**XID.** exchange ID.

## Numerics

**3270 data stream.** The commands, control codes, orders, attributes, and data or structured fields for 3270 devices that are transmitted between an application program and a terminal. Data being transferred from or to an allocated primary or tertiary device, or to the host system, as a continuous stream of data and 3270 Information Display System control elements in character form.





# Index

## Symbols

/etc/resolv.conf 98  
/usr/sbin/ftpd 91

## Numerics

3174 control unit 242  
3270 data stream 243  
3270 display terminal 243, 246

## A

abend 305  
abend (see abnormal end) 305  
abnormal end (abend) 304  
ACK flag 38  
address resolution protocol (ARP) 29, 43, 60  
address space 73, 95, 97, 106, 250  
    dump 307  
    TCP/IP 314  
adjacent link station (ALS) 146, 159  
Advanced Peer-to-Peer Networking 42, 136, 165  
Advanced Program-to-Program Communication 167  
alias address, VIPA 87  
American National Standard Code for Information (ASCII) 93, 102  
ANR 198  
APPC 167  
application program 11, 73, 135, 148, 157, 170, 244, 308  
application programming interface (API) 135, 167  
APPLS 310  
APPN 165  
APPN class 195  
APPN network 178, 191  
    central directory server 191  
    communications partner 168  
    defining 206  
    directory support 181  
    end node 179  
    network node 184  
    topology 184  
APPN node 141, 167, 208

    building block 169  
    control information 167  
    CP-CP sessions 177  
APPN traffic 61, 301  
architecture 74  
area border router (ABR) 36, 125  
area, OSPF 125  
ARP 29, 60  
ARP broadcast 30  
    area 29  
    capability 30  
    will 34  
ASCII 93  
associate 248  
Asynchronous Transfer Mode (ATM) 25, 135  
ATCSTRxx 155  
attention key 245  
authorized problem analysis report (APAR) 328  
automated teller machine (ATM) 8  
automatic network routing (ANR) 166, 207, 240  
availability 19

## B

backbone 13  
backup distributing host 116  
base control program (BCP) 91  
BEGINROUTES 89  
BEGINVTAM 253  
BEX 175  
BFRUSE 310  
bind  
    DNS 82  
    socket 106  
boundary function (BF) 140, 145, 178  
branch extender 175  
branch office 15  
BrEx 175  
broadcast search 188  
buffer 310

## C

capacity planning 268  
carrier-supplied communication line (CSCL) 15

- CCL 139
- CCW 52
- CDS 174
- central directory server 174
- central directory server (CDS) 192, 240
- central processor complex 46, 70
- central processor complex (CPC) 48, 70, 73
- CF 55, 73
- change control 268
- channel 48, 52
- channel command word (CCW) 52
- channel command word (CTC) 52
- channel interface processor (CIP) 55
- channel path identifier 51
- channel subsystem 51
  - main tasks 48
- channel-to-channel (CTC) 155
- character set 93
- CHPID 51
  - mapping tool (CMT) 48
- CICS 73
- CIPHERSUITE 288
- CISCO router 122
- class of service (COS) 155, 194
- CLIST 309
- cluster 104
- CNN 177
- command output 269, 311, 325
- Common Link Access to Workstation (CLAW) 55
- common service area (CSA) 320
- communication controller 139, 157, 167
  - of Linux 139
- communication line 139, 143, 196
  - communication controller 139
  - peripheral nodes 139
- Communications Server (CS) 15–16, 138, 148, 159, 249
  - SNA component 138
- Communications Storage Manager (CSM) 123
  - problem determination 324
- compatibility 74
- composite network node 177
- connection request 39, 43, 116, 119, 154, 294
  - specific port 154
- connectivity 46
- control information 158, 179, 244
- control point
  - network accessible unit 149
- control point (CP) 137, 152, 169, 180

- control unit
  - device type 242
- control unit (CU) 48, 242, 247
- convergence, network 123
- Coupling Facility (CF) 104
- coupling facility (CF) 55, 76, 78
- CPC 73
- CP-CP session 178, 183
- cross-domain resource manager
  - other domains 142
- cross-domain resource manager (CDRM) 142, 179
- cross-system Coupling Facility 105
- cross-system coupling facility 55
- CSM 324
  - address space 324
- CSS 46
- CU 51
- Customer Information Control System (CICS) 6, 78, 135

## D

- data link layer 25
- data protection 18
- data RUs 151
- data set 49, 155, 206, 250, 263–264, 306, 353
  - CTRACE 322
    - DD statements 305, 321
    - SYSMDUMP 306
- data stream xv, 244, 246, 290
  - fundamental characteristics 241
- data stream commands 244
- database
  - APPN 181
- Database 2 (See DB2) 73
- datagram 40–41, 85, 102, 317
  - forwarding 87
- DATAGRAMFWD 87
- DB2 73
- DD statement 84, 305
- DEFAULTTCPIPDATA 98
- defined network interface
  - device status 315
- dependent LU 148, 205
  - server 205
    - session initiation flows 205
- dependent LU requester/server 204
- dependent LUs
  - APPN support 165

- dynamic definition 159
- destination address field (DAF) 28
- destination host 34, 275
  - MAC address 34
- destination logical unit (DLU) 190
- development LPAR 72
- DEVICE 51
- device type 61, 74, 85, 114, 246, 248, 308
- direct access storage device (DASD) 51
- direct memory access (DMA) 67
- directed search 190
- director 51
- directory service 187
  - database 183
- DISPLAY NETSTAT 276
- DISPLAY TCPIP 102, 273, 316
- DISTMETHOD BASEWLM 122
- DISTMETHOD ROUNDROBIN 122
- distributed directory searches 181
- distributing host 116
- DMA 59
- documentation 280
- domain 141
- domain name services (DNS) 41, 96
- DOMAIN statement 96
- dump 305
  - FFST 308
  - stand-alone 306
  - SVC 306
- DVIPA 106
- DVIPA move 123
- dynamic definition 136, 159, 166, 208
- dynamic routing 35
- dynamic VIPA 128, 264
- dynamic virtual IP address 106
- dynamic VPN 291, 339
- dynamic XCF 113, 118

**E**

- EA S 257
- EBCDIC 93
- EBN 175
- either NN (EN) 207
- encryption 18
- end node 169–170, 179
  - CP-CP sessions 180
  - Multiple attachment points 171
- end user 4, 93, 148, 151–152, 197, 245, 295

- Enterprise Extender (EE) 12, 41, 74, 163, 264, 269, 301
- ephemeral port 297
- error recovery 196–197, 263
- ESCON channel 53
  - maximum channel data rate 55
- Ethernet 25
- exchange data 179
- explicit route
  - virtual route 144–145
- explicit route (ER) 144, 310
- Extended Binary Coded Decimal Interchange Code 101–102, 244
- extended border node (EBN) 165, 208
- Extended Recover Facility 62
- external communications adapter (XCA) 209
- extranet 5

## F

- Fast Ethernet 26
- FFST 308
- FFSTDF 309
- Fiber Distributed Data Interface (FDDI) 25
- fiber optic
  - cable 24, 53
- FICON channel 55
- FID2 150
- FID4 150
- field attribute 245
- File Transfer Protocol 65, 81, 92
- firewall 41, 43, 291, 297
- First Failure Support Technology (FFST) 328
- first-in-first-out (FIFO) 144
- flow control 41, 198, 209
- flow reduction sequence number 187
- form indicator 150
- forwarding agent 122
- FTP 90, 120
- FTP client 101, 119–120, 290
  - end 120
  - end users 288
  - issue 119
  - key ring 290
  - log 92
  - validate 290
- FTP server 39, 92–93, 120, 277, 288, 291
  - source code 289
- FTP session 92, 120

control connection 323  
function key 245

## G

gateway 42  
    function 140  
Gbps 24, 55, 58  
generalized trace facility (GTF) 312, 320  
Gigabit Ethernet  
    full duplex implementation 26  
GLOBALTCPIPDATA 98  
GTF 312

## H

HALT 272  
handshake, TLS 287  
hardware configuration  
    dialog 49  
    operation 46  
hardware configuration dialog 49  
Hardware Management Console (HMC) 46  
HCD 49  
High Performance Routing (HPR) 136, 196, 209  
HiperSockets 63, 114  
HMC 50  
HOME statement 85–86, 108, 276  
Host Integration Server (HIS) 148, 250  
HOSTNAME statement 96  
HPR 196  
HPR headers 200  
HTTP server 82  
    bind request 108  
    request 107–108

## I

I/O 51  
I/O cage 58, 65  
I/O configuration  
    data 49  
    program 49  
I/O Configuration Data Set (IOCDS) 49  
I/O definition file (IODF) 49  
I/O device 51  
I/O program (IOP) 58  
I/O request 48  
ICMP processing (IP) 27, 40  
ICN 176

IDS 42  
IEASYMxx 264  
IETF (See Internet Engineering Task Force) 28  
INACTIVE and PRTINACTIVE (IP) 250, 255  
inbound and outbound (I/O) 49, 52, 313, 320  
independent LU 149, 159, 205  
    Dynamic definition 159  
independent of physical (IP) 73–74, 78, 95, 103,  
105, 113, 139, 162–163  
inetd 94  
Information Management System (IMS) 19  
Interactive Problem Control System (IPCS) 313  
interchange node 176  
intermediate session routing (ISR) 166, 196  
internal QDIO 63  
Internet 5  
Internet Control Message Protocol 36  
Internet Engineering Task Force (IETF) 287  
Internet Key Exchange (IKE) 291  
inter-system coupling (ISC) 65  
intranet 5  
intrusion detection  
    capability 302  
    service 121  
IOCDS 49  
IOCP 50  
IODF 49  
IP address 29, 74, 86, 96, 118, 269, 298, 318  
    coming standard 83  
    eligible 108  
    new instance 112  
    normal owner 116  
IP application 28, 82, 98, 105, 321  
    huge number 82  
    IETF governs standards 28  
    operating characteristics 96  
IP communication 12, 89, 267  
IP filtering 249  
IP header 28  
IP layer  
    function 87  
    model 290, 292  
IP network xv, 12, 22, 31, 34, 106, 149, 199, 273,  
280, 296, 301, 328  
    layer 3 functions 31  
    SNA  
        traffic 269  
    status 296  
IP traffic 59, 112, 291

IPCONFIG 87  
IPSec 42, 290  
IPSec architecture 290  
IPv6 83  
IQDIO 63

## J

job control language (JCL) 82  
job entry subsystem (JES) 83  
JOBNAME 90, 315

## K

Key term xv  
KEYRING 289

## L

LAN 29  
LAN segment 29, 60  
    z/OS host 32  
last octette 33  
layer  
    data link 25  
    network 27  
    physical 23  
    transport 37  
LCSS 48, 51  
Lightweight Directory Access Protocol (LDAP)  
    policy definitions 291  
link configuration 84  
local area network (LAN) 11, 21, 29, 59, 63, 127,  
268  
local LU 181  
logical connection 145, 158  
logical partition 12, 46, 50, 70, 78, 105  
logical path 144  
logical unit 146  
    applications access SNA networks 146  
    different kinds 148  
    several types 204  
    SNA network 152  
logical unit (LU) 152, 179, 201, 248–249, 301, 321  
Lookat Web site 304  
low-entry networking 169  
low-entry networking (LEN) 183  
low-entry networking node 169  
LPAR 59, 70–71, 89, 104, 116, 267  
LPARs 46, 62, 76

LU 146  
    type 148  
LU 6.2 167  
LU name  
    selection 249  
LU-LU session 152, 175, 301  
LUs 147, 172, 248, 280

## M

MAC address 27, 61, 127  
    layer 2 30  
mainframe 5  
    connectivity 46  
mainframe computer 10  
mainframe network xv, 70, 177  
    capabilities 12  
    example 7, 14  
    IP 17  
    requirements 70  
    support 10  
MAJNODES 310  
major node 157, 311  
mapping 249  
maximum transmission unit (MTU) 274, 327  
Mbps 26, 65  
media access control (MAC) 27  
MIF 51  
migration data host (MDH) 209  
MODIFY 271  
MODIFY TRACE 309  
monitoring 262  
MultiNode Load Balancing (MNLB) 122  
MULTIPATH 87  
multiple application instance DVIPA  
    backup host 111  
multiple application-instance DVIPA 110, 129  
multiple image facility (MIF) 49, 51  
multi-stack environment 100  
MVS 91

## N

NAMESERVER statement 96  
NETACCESS 297  
NETID 141  
NETSTAT command 101, 273, 275, 296, 317  
NETSTAT CONFIG 316  
NETSTAT CONN 315  
NETSTAT DEV 315

- NETSTAT HOME 315
- NETSTAT ROUTE 315
  - command 275, 302
  - output 317
- NETSTAT SOCKETS 315
- NETSTAT STATS 315
- NETSTAT VIPADCFG 315
- network
  - availability 73
  - capabilities 12
  - examples 7
  - importance 7
  - LAN 29
  - layers 21
  - routing 31
  - security 17
  - types 29
  - virtual LAN 30
  - WAN 30
- network access 295
- network accessible unit 146
  - control communication 149
- network accessible unit (NAU) 149
- network administrator 10, 21, 101, 263, 273, 279, 327–328
  - quick way 279
  - responsibilities 20
- network control 268
- Network Control Program (NCP) 61, 139, 177, 209
- network ID 33, 113, 181, 255, 280
  - IP addresses 255
- network identifier 141
- network interface card (NIC) 13, 24, 27, 126
- network layer xiv, 21, 291
- network mask 33
- network node 171
  - APPN backbone 171
  - CP-CP connectivity 175
  - current connectivity 346
  - large number 12
  - transmission groups 183
- network node (NN) 178–179, 265, 318
- network node server (NNS) 170
- network problems 304
- network resource 137, 140, 179, 294
- network security 297
- network shutdown 267
- network startup 263
- network topology 102, 173, 187

- efficient use 88
- network traces 309
- NFS 82
- NIC 24
- node type 169
- node, NODETYPE (NN) 177
- nodes 138
- nondisruptive path switching 201

## O

- OBEYFILE command 275, 282
- OLE\_LINK1 137
- OLTP 6
- OMPROUTE 76, 124
- online system 5
- online transaction processing 6
  - system 6
- Open Shortest Path First (OSPF) 43, 73, 76, 124
- Open Systems Adapter (OSA) 23, 56, 73, 143, 145, 205
  - for NCP 61
  - main versions 56
- Open Systems Adapter (OSA) (see OSA) 56
- operating system xiii, xvi, 6, 26, 49, 100, 263, 374
  - on-line help xvi
  - working knowledge 10
- operation-request block (ORB) 53
- operations 261
- origin address field (OAF) 150
- origin LU 193
- originating logical unit (OLU) 182
- OSA card 24, 56, 76, 87
  - MAC addresses 27
- OSA card (see open systems adapter) 73
- OSA-2 card 56
- OSA-E, sample TCP/IP definitions 84
- OSA-Express 126
- OSA-Express 2 card 56
- OSA-Express card 13, 56, 58, 89, 126
  - availability features 127
- OSA-Express2 card 27, 58
- OSN 61
- OSPF 35, 73, 124
- OSPF area 125
- otelnetd, z/OS UNIX Telnet server 94

## P

- packet trace 296

- paperwork 280
- parallel 52
- parallel channel 24, 52, 139
- Parallel Sysplex xvi, 12, 104
- partner LU 152, 201
- PATH DESTSA 160
- path information unit (PIU) 150, 162, 200
- peer-to-peer networking 168
- performance 326
- peripheral node 139, 158, 167
- personal identification number (PIN) 8
- physical unit 146
  - dynamic definitions 208
- physical unit (PIU) 149
- physical unit (PU) 157, 269
- PING 316
- ping command 101
- planning 268
- point-of-sale (POS) 7
- policy agent 121, 298, 302
  - full description 121
  - other advantages 293
- port 39, 249
- port access 295
- port number 39, 41, 95, 106, 297, 323
- PORT statement, TCP/IP profile 298
- PORTACCESS 297
- PORTRANGE 297
- predefined using a switched (PUS) 157, 194
- primary logical unit (PLU) 153, 183
- printer LU 257
- PRIROUTER, DEVICE parameter 85
- production LPAR
  - production sysplex 72
- profile statements 84
- program temporary fix (PTF) 328
- program update tape (PUT) 307
- protocol 11
- PU 146

## Q

- QDIO 58
- QDIO (See Queued Direct I/O) 59
- QDIO mode 60
  - OSA-Express port operating 59
- Queued Direct I/O (QDIO) 57

## R

- rapid transport protocol (RTP) 197, 270
- RAS 74
- receive buffer size 88
- recovery 122
- Redbooks Web site
  - Contact us xvii
- reliability, availability, serviceability (RAS) 78
- remote host 38, 273, 322, 324
- request header
  - response header 152
- request header (RH) 151
- request unit (RU) 151, 192, 200
- request/response indicator (RRI) 152
- resolver 95
- RESOURCE 50–51
- Resource Access Control Facility (RACF) 18, 286
- resource sequence number (RSN) 186
- response header (RH) 151
- response unit (RU) 152
- RESTRICTLOWPORTS 297
- RFC 28
- RFC 4217 288
- RIP 35
- RIP (Routing Information Protocol) 123
- rlogin 82
- round trip time (RTT) 327
- ROUTE 310
- route calculation 193
- route entry 317
  - destination field 317
- route extension 144
- router 55, 76
- routing 123
  - dynamic 35
  - general 31
  - static 34, 88
- Routing Information Protocol (RIP) 35
- Routing Information Protocol (RIP) 123
- routing table 32–33, 126, 199
- RTP 197
- RTP connection 198
  - endpoints 199

## S

- same subnetwork 114
  - unique IP address 113

- scalability 74
- SDSF 310
- SDUMP 306
- SE 50
- search order 99
- secondary logical unit (SLU) 153, 182
- Secure Sockets Layer (SSL) 42, 249, 287
- SECURE\_CTRLCONN 289
- SECURE\_DATACONN 289
- SECURE\_FTP 288
- SECURE\_LOGIN 288
- security 41, 74
- security policy 41, 281, 284
- Self-Timed Interface (STI) 58
- sendmail 82
- sense codes 309
- sequence number 38
- service access point (SAP) 42
- service level agreement (SLA) 75, 121
- session 141
- session continuity 166, 197
- session partner 153, 170, 301
  - available route 166
  - response unit flows 153
- session path 195
- session-initiation request 152, 155
- Simple Mail Transfer Protocol (SMTP) 15
- single LPAR 55, 64, 116, 295, 299
  - multiple Linux images 55
  - separate IP stacks 299
- SLA 75
- SLIP 306
- SMS (system management services) 310
- SNA application 15, 135, 246, 248, 257
  - program 157
- SNA message 149
- SNA network 22, 61–62, 138, 146, 152, 154, 175, 204, 242, 250
  - architectural components 146
  - interconnection 175
  - major components 133
  - stores text definitions 155
- SNA Network Interconnect (SNI) 62
- SNA over IP 136
- SNA resource 42, 158, 166
  - static definitions 166
- SNA session 136, 182, 198, 249, 254
  - side 252
  - terminal LU 257
- SNA subarea cross domain network 143
- SNA subarea network 152, 166, 209
- SNA traffic 41, 300
  - security requirements 300
- SNA/IP Implementation 131
- SNMP 82
- snmp command 101
- socket 40
  - programming 135
- SSCP 141
- SSCP-LU session 147, 205
- SSH 82
- SSL 42
- stack access 295
- STACKACCESS 297
- stand-alone dump 306
- started task 82
- static route 34, 89, 317
- static routing 34
  - big advantage 88
- stub area 78, 125, 129
- subarea definitions 155
- subarea network 140–141, 145, 154, 159, 162, 166, 175
  - data host 179
  - network resources 141
- subarea node 42, 143, 150, 161–162, 169, 179
  - logical connection 143
  - ordered set 144
  - same set 144
  - SSCP-SSCP sessions 177, 179
- subnetwork 89, 175
- superuser 295
- support element (SE) 50
  - image profile 50
- SVC 306
- switched major node 157
- switched PUs
  - message exchange 159
- symmetric encryption 301
- SYS1.VTAM LST 158
- SYSABEND 305
- SYSDEBUG 321
- SYSERR 321
- SYSERROR 321
- SYSMDUMP 305–306
- sysplex 72, 103
- sysplex distributor 103, 117, 128
  - environment 264

- function 122
- host 116
- node 122
- SYSPLEXMONITOR 123
- SYSPRINT 321
- SYSTCPD DD
  - card 99
  - DISP 83
- SYSTCPDA 314
- SYSTCPIP 314
- SYSTCPIS 314
- SYSTCPRE 314
- SYSTCPRT 314
- system assist processors 46
- system authorization facility (SAF) 74, 249, 286, 302
  - security capabilities 286
- system display and search facility (SDSF) 350
- system services control point (SSCP) 148, 178
- Systems Network Architecture
  - history 134
  - layers 42
  - sample network 159
  - subarea networking 134
- Systems Network Architecture (SNA) 9, 11, 43, 163, 242, 353
  - nodes 138
- systems network architecture (SNA) 61, 300–301
- SYSUDUMP 305

**T**

- target host 36, 116–117, 294, 320
- target server 121
- TCP 37
- TCP application 39, 88
- TCP connection 39, 116, 273
  - possible states 39
- TCP/IP 16–17, 63, 73–75, 78, 81–82, 87, 97, 101–102, 117, 121, 242, 250, 286, 293–294, 297, 348
  - address space 320–321
  - clients 101
  - displaying 273
  - problem determination 320
  - sysplex point-to-multipoint network 113
- TCP/IP applications, controlling 277
- TCP/IP component trace 313
- TCP/IP profile 83
  - data 84, 102, 113, 292
  - statement 84, 108
- TCP/IP startup 265
- TCP/IP task 82–83, 265
- TCPCONFIG 87
- TCPIP 83, 95, 266
- TCPIPJOBNAME statement 96, 295
- TCPMAXRCVBUFRSIZE 88
- TCPRCVBUFRSIZE 88
- TCPSENDERBUFRSIZE 88
- telnet 94
- Telnet 3270 242, 246
- telnetd 94
- TELNETPARMS 251
- terminal LU 248, 257
- terminal session protocol (TSP) 279
- test LPAR 72
- TGET/TPUT 310
- Time Sharing Option (TSO) 73, 91, 101
- TLS 249
- TN3270E client 247
  - communicate 247
  - window 248
- TN3270E connection 252–253
- TN3270E server 248, 254, 256, 266, 277
  - basic configuration elements 241
  - secure data transmission 249
  - SNA session side 252
  - TN3270E protocol ends 249
  - VTAM setup 256
- TN3270E servers 250
- topology 11
- topology and route selection services 181
- topology data update (TDU) 184, 186
- topology database 177, 182, 185, 193
  - central directory server 175
  - node record 186
- traceroute 101
- TRACERTE 316
- tracerte 101
- transit area, OSPF 125
- Transmission Control Protocol (TCP) 37, 43
- transmission group
  - characteristic 194
  - number 143
  - profile 194
  - vector 193
- transmission group (TG) 143–144, 182, 186, 197
- transmission header (TH) 150, 200, 351

transmission priority (TP) 144, 154, 200  
transport layer 37  
Transport Layer Security (TLS) 19, 42, 249, 287,  
302  
transport resource list (TRL) 310–311  
TRL 310  
trouble-shooting 269  
TSO 73, 310  
TTLS 292  
tuning 326  
Type of Service (TOS) 154

## U

UDP 40  
unique application-instance DVIPA 106  
UNIX System Services 91  
unrepeated distance 55  
UNSECURE Maximum 195  
Unshielded Twisted Pair (UTP) 24  
User Datagram Protocol (UDP) 40, 101  
user ID 60, 99, 273, 287, 296  
USR sense 0801 309  
USR sense 08010010 309  
USS message 253  
table 253

## V

VARY 271  
VIPA 74, 86  
VIPA address 77, 86–87, 102, 108, 110  
VIPABACKUP 111  
VIPADefINE 111  
VIPADYnAMIC 108  
VIPARANGE 108  
virtual IP address 74, 86  
virtual LAN 30  
virtual private network (VPN) 19, 121, 280, 290,  
302  
virtual route 145, 163, 310  
Virtual Telecommunication Access Method (VTAM)  
xvi  
Virtual Telecommunications Access Method  
(VTAM) 16, 77–78, 105, 134, 138, 169, 178,  
208–209, 266, 271, 279, 310  
VLAN 30–31, 67  
VLAN ID 31  
voice data 4  
VTAM 16, 73, 138

address space 320  
application program 308  
buffer content 313  
buffer usage 325  
changing resources 271  
component trace 313  
diagnostic tools 308  
displaying resources 269  
I/O trace 310  
internal trace 319  
problem determination 318  
start option 310  
VTAM buffer  
trace 272  
usage D Net 310  
VTAM DISPLAY Commands 310  
VTAM major node 206, 311  
VTAM resource 158, 208, 271  
VTAM start  
option 156, 272  
VTAM startup 264  
VTAM traces 310  
VTAMLST 256  
VTAMOPTS 310

## W

WAN 30  
Web site 28, 78  
Web Site Voice 135  
wide area network 11, 21, 30  
window size 88  
Workload Manager (WLM) 105  
write control character 244  
write control characters (WCC) 244  
Write Structured Field (WSF) 244

## X

XCF 55, 104–105  
XRF 62

## Z

z/OS 64, 82, 98, 117, 122, 138, 155, 249, 263, 277,  
328  
z/OS Communications Server 16, 249–250  
components 15  
DNS-based functions 250  
IP Configuration Guide 90

- SNA Operation 269
- z/OS environment 74, 82, 269, 294
- z/OS host 32, 52, 98, 121, 124, 249, 268, 284
  - IDS functions 294
  - TCP application 88
- z/OS image 105, 264
- z/OS IP
  - Security Configuration Assistant 292
  - System Administrator 273
- z/OS operating system xiv, 12, 49, 73, 91, 104, 128, 298
  - Communications Server 15
    - distinct instances 104
    - tightly-coupled independent instances 128
- z/OS system 61, 73, 101, 148, 155, 273
  - DUMP command 321
  - interactive session 73
  - log 322
  - programmer 22
- z/OS System Services 91
- z/OS UNIX 91–92, 124, 295–296
  - application 91
    - search 98
  - command 101
  - environment 273
  - environment form 91
  - file 94
  - fundamental difference 93
  - HFS 93
  - hierarchical file system 92
  - service 91
  - snmp command 101
  - Telnet server 94
- z/OS UNIX System Services 91





## Introduction to the New Mainframe: Networking

(0.5" spine)  
0.475" <-> 0.875"  
250 <-> 459 pages







**Redbooks**

# Introduction to the New Mainframe: Networking

**Concepts of mainframe-based data communications**

**Methods for implementing TCP/IP and SNA on z/OS**

**Basic skills in network operations, security, and problem determination**

This IBM Redbook is designed for a one-semester course intended to provide information systems students with the background knowledge and skills necessary to begin using the basic communications facilities of a mainframe-based system. It provides a broad understanding of networking principles and the hardware and software components necessary to allow the mainframe to participate in a high volume data communications network. It is part of a planned series of textbooks designed to introduce students to mainframe concepts and help prepare them for a career in large systems computing.

While many of the networking concepts covered are operating system-independent, the main emphasis is on the z/OS operating system. You are assumed to have successfully completed introductory courses in computer system concepts, including courses in computer organization and architecture, operating systems, data management and data communications, and systems design and analysis.

A basic understanding of z/OS job control, library structure, and system libraries is assumed. It is strongly recommended that you have already completed an introductory course on z/OS, such as one that uses *Introduction to the New Mainframe: z/OS Basics* or a comparable text. In some instances, the *z/OS Basics* course and the course associated with this textbook could be taken together.

## **INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION**

### **BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**  
[ibm.com/redbooks](http://ibm.com/redbooks)