CA XCOM™ Data Transport® for HP NonStop

Product Guide

r11

CA Technologies Product References

This document references the following CA Technologies products:

- CA ACF2® for Security (CA ACF2)
- CA Dynam®/T Tape Management (CA Dynam/T)
- CA Top Secret® Security (CA Top Secret)
- CA XCOM™ Data Transport® for HP NonStop (CA XCOM Data Transport for HP NonStop)

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Introduction 19

Chapter 4: Configuring the Software 79

Chapter 5: The Batch and Command Line Interface 143

Chapter 1: Introduction

This chapter introduces CA XCOM Data Transport. Read this chapter before installing or configuring CA XCOM Data Transport.

This section contains the following topics:

Product Overview

CA XCOM Data Transport is a family of software products that operates under SNA using LU 6.2, or under TCP/IP, to provide high-speed data transfer between supported systems such as mainframes, midrange, PCs, servers, and workstations. You can send files from their local system to remote systems across an SNA network or using TCP/IP, and retrieve files from those remote systems. The same transfer capabilities are available to the local and remote systems.

CA XCOM Data Transport provides a unified solution for communications over more operating systems than any other software product on the market today. CA XCOM Data Transport also has a solid technology base. By using LU 6.2 or TCP/IP communications protocols, CA XCOM Data Transport uses state-of-the-art technology, protecting your company's investment for years to come.

How the Data Transport Process Works

To understand the data transport function in a very simplified and generalized way, consider a scenario. For example, when a local system transfers a file to a remote (partner) systems, following steps are performed:

1. Initiation

   The user submits a batch program, starts the menu (the menu interface) or a customer program written using the XCOM API (application programming interface) to initiate the transfer.

2. Information verification

   CA XCOM Data Transport verifies the information contained in the request. For example:

   - When requesting a send file transfer, CA XCOM Data Transport checks whether the local file exists on the local system.
   - When requesting a receive file transfer, CA XCOM Data Transport checks whether the file exists on the remote system.

3. Information confirmation

   If the information is confirmed, CA XCOM Data Transport starts the file transfer.

4. Completion

   The transfer completes and CA XCOM Data Transport logs the details of the transfer in a log.

Note: The previous example illustrates a general idea of how CA XCOM Data Transport works, please be aware that it is simplified; there are many more steps involved in the process. For more information about how data transport works, see the other chapters in this guide.

Summary

CA XCOM Data Transport is a widely used, proven vehicle for moving data between a growing numbers of dissimilar systems. CA XCOM Data Transport provides security, recovery, scheduling, and administrative facilities.

By using CA XCOM Data Transport, you can realize the following advantages:

- Effectively utilize the existing investment in data processing hardware.
- Reduce costs by replacing multiple information transfer products with a single, easy-to-use package.
- Reduce operations and end-user staff training costs by implementing a centrally-controlled, highly automated data transfer solution.
- Provide an environment that supports the development of strategic new applications.
- Increase the flexibility and accessibility of remote systems, allowing your organization to respond quickly and accurately to changing business needs.

Applications You Can Design Using the Product

CA XCOM Data Transport allows data centers in various locations worldwide to interact with each other for the following purposes:

- Sharing data
- Automating data and report distribution
- Providing unattended back-up to dissimilar computers
- Controlling and auditing network activities
- Maintaining network security
- Communicating with Point-of-Sale (POS) terminals

The applications listed above are only a few examples. Under most conditions, CA XCOM Data Transport allows file sharing between any two systems or workstations within your company.

Applications Using the Transfer Function

The key to the considerable flexibility of CA XCOM Data Transport is its ability to transfer the following:

- Files
- Jobs
- Reports

When these functions are combined, a wide variety of applications are possible.

Types of Transfers

CA XCOM Data Transport performs the following transfers:

Sending Files

With CA XCOM Data Transport, a local system can send a data file to be stored on the remote system in a specified remote file.

Sending Reports

CA XCOM Data Transport can send a report to be printed on a remote system.

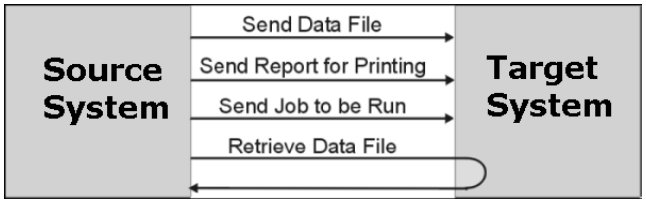
Sending batch jobs for execution

CA XCOM Data Transport can send a job to be executed on a remote system.

Retrieving files

When a system starts the transmission request, it can also retrieve a file from a remote system and store it in a specified local remote file.

The following flow chart illustrates the type of transfers supported by the product:



How Remote Requests Are Handled

You can use CA XCOM Data Transport to monitor the network for incoming requests. Upon detecting one, CA XCOM Data Transport determines whether it is a request to send a file inbound (from the remote system to this system) or outbound (from this system to remote system).

File Transfers

You can use the file transfer feature to send or retrieve files from a remote system to a local system.

When CA XCOM Data Transport transfers a file from a remote system to your system, the following actions occur:

- The remote system requests CA XCOM Data Transport to send a file to your system.
- CA XCOM Data Transport allocates memory to the requesting process and opens the file.
- CA XCOM Data Transport then reads the data records from the file.
- CA XCOM Data Transport transfers the file to your system.
- Your system receives the file.

Job Transfers

When CA XCOM Data Transport transfers a job from a remote system to your system, the following actions occur:

- The remote system requests CA XCOM Data Transport to submit a job to your system.
- CA XCOM Data Transport submits the job to your system.
- Your system receives the job file.

Report Transfers

The report transfer feature allows a remote system to send a report to a local system. CA XCOM Data Transport provides a high degree of print redirection and spooling capabilities.

When CA XCOM Data Transport transfers a report from a remote system to your system, the following actions occur:

- The remote system requests CA XCOM Data Transport to send a report to your system
- CA XCOM Data Transport writes the report to an output spool file.
- CA XCOM Data Transport transfers the file to your system.
- Your system retrieves the report from the spool file.

CA XCOM Data Transport Features

CA XCOM Data Transport provides peer-to-peer communications using LU 6.2 or TCP/IP over a wider range of systems than any other product. All of the major features of CA XCOM Data Transport are supported across the product line.

File Transfer

CA XCOM Data Transport supports high-speed file transfers between all supported operating systems. In some environments, you can start thousands of transfers resulting in hundreds of simultaneous transfers, all with a single operation. Parallel sessions are possible in varying degrees throughout the product line.

You can totally automate CA XCOM Data Transport transfers. On a PC, you can be actively engaged in the use of other applications (for example, word processing) while receiving or transmitting files in the background. Comprehensive management tools allow for effective central-site control of CA XCOM Data Transport activity, including advanced problem determination features.

CA XCOM Data Transport supports transfers between any two systems in an SNA network or a TCP/IP network with one of the following methods:

- By using the z/OS, z/VM, or z/VSE mainframes for store-and-forward
- Through Independent Logical Unit (ILU) support over the SNA (Systems Network Architecture) backbone
- Through use of the TCP/IP network (except for z/VM and Stratus)

Type 2.1 Support

CA XCOM Data Transport supports node Type 2.1 connections to allow the direct interchange of files between the Windows operating environment, NetWare workstations, and others. Support for Independent Logical Units (ILUs) allows CA XCOM Data Transport to deliver data in Advanced Peer to Peer Networking (APPN) and Low Entry Networking (LEN) networks. This means that PCs and midrange attached to the same SNA or APPN network can exchange data even if they are not directly connected.

TCP/IP Support

CA XCOM Data Transport provides support for performing transfers using TCP/IP between CA XCOM Data Transport platforms that support TCP/IP and that are running r3.0, r3.1, r11, or r11.5. TCP/IP support is provided between the following platforms:

- i5/OS (AS/400)
- Linux S/390
- Linux x86
- NetWare
- Open VMS Alpha
- HP NonStop (Tandem)
- z/VSE
- Windows family
- z/OS
- Most common UNIX platforms

You can use the Secure Socket Layer (SSL) to perform secure TCP/IP transfers between platforms running CA XCOM Data Transport r11 and above that have this support enabled. CA XCOM Data Transport uses OpenSSL to encrypt the transmitted data and adds a digital signature to the encryption of the transmitted data. Secure TCP/IP support is provided between the following platforms.

- i5/OS(AS/400)
- Linux S/390
- Linux x86
- Windows family
- z/OS
- Most common UNIX platforms

Report Distribution

CA XCOM Data Transport allows z/OS, z/VM, z/VSE, i5/OS (AS/400), and OpenVMS Alpha users to take print output from any supported system and automatically transfer it to another system for printing. The application programs producing the reports do not require any modification to support CA XCOM Data Transport report distribution, and no operator intervention is required at either end.

RJE Replacement

Current Remote Job Entry (RJE) systems contain inherent limitations. Remote systems can submit work to the host for processing and receive print data, but the host cannot distribute processing tasks to idle processors residing on the network. A further concern for data processing managers is the requirement that users are familiar with Job Entry Subsystem (JES) commands to operate the system.

CA XCOM Data Transport avoids these limitations by taking advantage of the LU 6.2 and TCP/IP protocols, providing a peer-to-peer relationship between all supported systems. Any CA XCOM Data Transport system is able to send and receive batch jobs and print data from any other CA XCOM Data Transport system without formatting constraints.

For example, an i5/OS (AS/400) user can do the following:

- Automatically retrieve files from a number of attached PCs.
- Process the data.
- Generate a report.
- Send one copy of the report back to the source PC for printing.
- Send another to the z/OS mainframe for printing on a high-speed printer.

You can easily implement CA XCOM Data Transport without any changes to your existing application programs. Data is transferred with greater integrity and higher efficiency.

Support of Most Operating Systems

By supporting the LU 6.2 and TCP/IP protocols, CA XCOM Data Transport can transfer data between a diversity of platforms. CA XCOM Data Transport is now available on the following systems now:

- Digital UNIX
- HP NonStop (Tandem) and TRU64 UNIX
- HP-UX
- IBM AIX
- i5/OS (AS/400)
- Linux S/390
- Linux X86
- MS Windows
- NCR 3000 (AT&T)
- Novell NetWare
- OpenVMS Alpha
- SCO OpenServer
- SCO UnixWare
- Stratus VOS
- Sun Solaris
- z/OS
- z/VM
- z/VSE

Data Link Types

CA XCOM Data Transport supports the following data link types:

- SDLC
- X.25
- Local Area Network (such as Token Ring and Ethernet)
- All SNA data links, including channel-based links
- TCP/IP

Standard Features

The following features are standard to CA XCOM Data Transport:

- Simple installation

  You can install CA XCOM Data Transport without hardware changes to your system.

- Initiation by either system (any-to-any)

  Either system can send and retrieve data files.

- Low maintenance

  There are no hooks or patches into the operating system.

- Choice of interface

  You can choose from batch/command line, programming (on supported platforms), and menu interfaces.

Standard Functions

The following functions are offered over most of the CA XCOM Data Transport platforms:

- Compression

  CA XCOM Data Transport offers a wide range of compression options on most platforms.

- Packing

  CA XCOM Data Transport can pack records into fixed-size data transfer blocks as large as 31K, significantly improving performance and throughput.

- ASCII/EBCDIC translation

  CA XCOM Data Transport can translate data between ASCII and EBCDIC formats as needed. Translations occur on the ASCII-based platform.

- Checkpoint/Restart

  All components of CA XCOM Data Transport support checkpoint/restart. Transfers that are stopped or fail prior to completion automatically resume, continuing from the last checkpoint.

- Store-and-forward

  Users communicating through a common z/OS, z/VM, or z/VSE hub can perform data transfers even if the remote (target) machine is not communicating or turned on at the time of the initial transfer. CA XCOM Data Transport ensures that the data is sent as soon as the device is available.

- Remote spooling

  CA XCOM Data Transport allows z/OS, z/VM, z/VSE, i5/OS (AS/400), and Open VMS Alpha users the following reporting options:

  - CA XCOM Data Transport on all platforms can receive reports.
  - CA XCOM Data Transport on all platforms can send a file to a remote CA XCOM Data Transport partner, requesting that it be treated as a report.
  - Some CA XCOM Data Transport platforms can also take reports off the system spool and forward them to another CA XCOM Data Transport platform without operator action. This automatic report transfer facility is called Process SYSOUT on z/OS and z/VSE, and it is called XQUE on AS/400, HP NonStop (Tandem), and Open VMS Alpha. The z/VM platform does not allow automatic processing of spooled files. However, spooled files on z/VM can be manually received and redirected.

High Capacity and Performance

CA XCOM Data Transport is optimized for high-speed bulk data transfer. For instance, CA XCOM Data Transport for z/OS can allow hundreds of simultaneous file transfers from a single system, depending upon your hardware and software configuration. Comparatively, CICS-based products limit the user to a maximum of 34 simultaneous transfers, and many other VTAM file transfer products are faced with similar limitations.

Security

CA XCOM Data Transport interfaces with the native security facility on all supported systems. When security is invoked, you are required to provide a valid user ID and password for the remote system. For example, in the z/OS environment, an interface is also provided to IBM RACF, CA ACF2, and CA Top Secret. Unlike most other communication facilities, CA XCOM Data Transport encrypts passwords. This ensures that communications line tapping does not breach security.

CA XCOM Data Transport also has special security capabilities that can help data centers handle their individual needs. For example, the security features of CA XCOM Data Transport allow installer specification of what can or cannot run under the privileges of someone other than the person requesting the transmission. These security features can also force user IDs from both remote systems to be the same or different. For otherwise unsatisfied security needs, CA XCOM Data Transport supplies a variety of user exits, which enable user-written security packages to be fully integrated.

CA XCOM Data Transport r11 (on some platforms) and r11.5 can also use the Secure Socket Layer (SSL) to perform data transfers under TCP/IP. CA XCOM Data Transport provides certificate authentication, data encryption, and data integrity ensuring all data transfers using SSL are secure.

Management

An important feature for any enterprise-wide information product is the ability to effectively control and manage the distribution of files and work throughout the network. CA XCOM Data Transport systems maintain a comprehensive log of all transfer activity. Utilities are provided to allow the system administrator to view the log online and modify the status of pending or currently active transfers.

Details of any transfer errors are also maintained in the log, allowing rapid problem determination and resolution. In addition, messages signaling the completion of any CA XCOM Data Transport event can be directed to a user in the network.

Invoking CA XCOM Data Transport

CA XCOM Data Transport is both easy to use and diverse enough for the most complex applications. Data transfer can be completed through any one of three interfaces, and reports can be printed on a remote printer using the Remote Spooling feature. These are described fully in later chapters.

Menu Interface

The menu interface provides a simple, fill-in-the-blanks approach to file transfer. You are prompted for required information and can use the extensive on-line help facilities provided with each product. CA XCOM Data Transport menus always have the look and feel appropriate to the system on which they are running. For example, the microcomputer platforms use graphical user interfaces with menus, tool bars, pop-up windows, and so on, while z/OS is written for the popular ISPF Dialog Manager.

Batch/Command Line Interface

CA XCOM Data Transport can also be initiated with a batch file on your computer. For example, a transfer can be invoked through a JCL batch job on z/OS and a CMS EXEC on z/VM. On mini and microcomputers, a transfer can be initiated via a command entered at the operating system prompt/command line or placed in a batch file to be executed with other commands.

Programming Interface

Any programming language supporting callable subroutines can call CA XCOM Data Transport. Examples of calling programs from common programming languages are given in each user manual, for supported platforms. CA XCOM Data Transport also provides exits on many systems that allow you to control or be informed about certain CA XCOM Data Transport events involving security and completed transfers.

Network Levels in the SNA Model

An SNA network is divided into levels of physical and logical components. A path control network that runs over the physical components interconnects the logical components.

As data is passed up and down the SNA functional layers, each layer performs a set of control functions and adds control information to the data in the form of a header. The headers do not change the information in the data, but communicate with the next layer of SNA to ensure that the data is understood.

As the data is passed through the layers, headers that are added at one end of the network are stripped off and read by the receiving end. Thus, when the data reaches its final destination, it is back to its original form.

The End User Level

The end user level consists of transaction programs (like CA XCOM Data Transport) that communicate with other transaction programs using LUs. In a CA XCOM Data Transport transfer request, the local end user specifies the following:

- Type of transfer (send report, file, or job, or receive file)
- Name of the local file
- Name of the remote file to create, append, or replace

The request is then processed by the CA XCOM Data Transport transaction program on the local system which sends an allocate verb and header record to the remote system to establish an LU 6.2 session. When a session is established and all of the parameters are confirmed by the remote system, CA XCOM Data Transport on the local system will send the file, broken into data records, across the physical connection.

The remote CA XCOM Data Transport transaction program is then invoked by the allocate verb sent by the local system (some systems cannot be automatically invoked and must already be active to receive the allocate verb). The remote transaction program then creates or opens the requested file, receives the data records, and places them in the file. When all the records have been received, CA XCOM Data Transport on the remote system will send a trailer record to the local system stating the number of records transmitted.

The Logical Level

The logical level consists of logical units (LUs), which link the physical units (PUs) and transaction programs (TPs). Each CA XCOM Data Transport user should be assigned a unique LU name for gateway or workstation identification. Each data record traveling across the line will contain LU 6.2 protocol information containing instructions for the remote system.

The Physical Level

The physical level consists of physical unit (PU) nodes linked by a physical connection. CA XCOM Data Transport supports transfers between physically connected systems whether they are directly or indirectly connected. PU Type 2.1 nodes can make a direct logical link even without a direct physical connection. Transfers involving PU Type 2.0 nodes (like VAX/VMS) can be made using the store-and-forward feature of CA XCOM Data Transport.

Benefits of LU 6.2 (APPC)

CA XCOM Data Transport is built upon the LU 6.2 protocol. LU 6.2 is also known as Advanced Program-to-Program Communications (APPC) and is IBM's most powerful enhancement to the Systems Network Architecture (SNA) suite of communications protocols.

Improved Throughput

When CA XCOM Data Transport uses the LU 6.2 protocol, it does not place hardware restrictions on the size of the data that it sends. RJE-based data transfer packages (LU Type 1) limit the size of a data frame (RU) to 80 characters, while 3270-based transfer packages (LU Type 2) often limit the size to 1,920 characters, the size of one screen. LU 6.2 allows RU sizes of up to 65,536 characters.

Each transmitted string of data is wrapped in a protocol envelope. Larger RU sizes, such as those allowed with CA XCOM Data Transport, mean less protocol overhead and a higher percentage of actual data traveling across your communication links.

Note: P = Protocol Overhead

Other LUs (RJE, and so on)

P	Data	**P**	Data	**P**	Data

LU 6.2 (CA XCOM Data Transport)

P	Data

Support for Advanced Networking

For SNA, LU 6.2 fully exploits the PU Type 2.1 peer protocol. This means that LU 6.2 is the only protocol that can use advanced functions such as Low Entry Networking (LEN), Advanced Peer-to-Peer Networking (APPN), and Independent Logical Units (ILUs) that are changing the face of computer networking today. If you are not familiar with these topics, you will find additional information in the appendix "About Logical Units."

APPC allows two programs running on distinctly different computers to converse with each other in real time without regard to hardware. All other protocols assume that one of the two devices communicating is a dumb terminal and impose all the limitations of a particular terminal on the partner computer. APPC and CA XCOM Data Transport recognize that distributed processing employs intelligent processors so they can exploit the intelligence of the computers on which they are running.

CA XCOM Data Transport in the TCP/IP Network

This section provides a discussion of the architectural and conceptual framework of the TCP/IP communications protocol as it relates to the implementation of CA XCOM Data Transport as a TCP/IP application.

TCP/IP Protocol Stack

TCP/IP is a collection of specialized communications protocols and functions organized into a stack of four layers. The layers that make up the TCP/IP protocol stack are the following:

- Network layer (the lowest protocol layer)
- Internetwork layer
- Transport layer
- Application layer (the highest protocol layer)

Each layer in the TCP/IP protocol stack provides services to the layer above it and uses the services below it.

The table below lists the protocols and functions that form the content of each layer of the TCP/IP protocol stack (the table shows only partial contents for the top and bottom layer).

TCP/IP Protocol Layer	Protocols and Functions
Network layer	Token Ring Ethernet X.25 etc.
Internetwork layer	Internet Protocol (IP) Control Message Protocol (ICMP) Address Resolution Protocol (ARP)
Transport layer	Transmission Control Protocol (TCP) User Datagram Protocol (UDP)
Application layer	Telnet File Transfer Protocol (FTP) Simple Mail Transfer Protocol (SMTP) Domain Name System (DNS) Sockets etc.

The next few sections provide (1) a summary of the services that each layer of the protocol stack performs in the TCP/IP network and (2) a description of the protocols that are particularly important for the functioning of CA XCOM Data Transport as a TCP/IP network node.

Network Layer

The network layer provides a set of protocols, Token Ring, Ethernet, and so on, that define how data are transported over different physical networks.

Internetwork Layer

The protocols of the Internetwork layer provide connection services for TCP/IP. The protocols in this layer connect physical networks and transport protocols.

The **Internet Protocol** (IP) of this layer integrates different physical networks into a unified logical network known as the Internet and provides for the universal addressing of computers (hosts) in a TCP/IP (internet) network. IP uses a 32-bit number (IP address) that identifies both a physical network and a specific computer within that network. The IP address is the basic transport mechanism for routing data from a source computer to a destination computer.

However, IP does not ensure a reliable communication, because it does not require that the computers participating in a data exchange acknowledge the reception of the transmitted data. The reliability of communication is implemented at the next higher protocol layer.

Transport Layer

The protocols of the Transport layer allow communication between application programs.

The **Transmission Control Protocol** (TCP) of this layer establishes a connection between the sender and receiver and provides a continuous communication service with reliability of transmissions. TCP divides the data to be transmitted into smaller units (packets, datagrams), sends them individually using IP, and reassembles them at the destination node, comparing the received data with the data that were sent. TCP is capable of determining if the two are the same. If they are not (data were lost or damaged during transmission), TCP resends the missing data.

Application Layer

The Application layer, which is built on the services of the Transport layer, provides a number of applications that allow users to use network services (terminal-to-terminal communication, data transfer, electronic mail, and so forth).

The Application layer provides an application-programming interface known as Sockets for communications applications. CA XCOM Data Transport uses this component of the Application layer to transfer files to machines in a TCP/IP network.

CA XCOM Data Transport Nodes in the TCP/IP Network

Each computer in a TCP/IP network is assigned at least one unique address, which is used by the IP and other higher-level protocols.

TCP/IP Address

TCP/IP employs an addressing scheme consisting of a 32-bit long field divided into two parts. The first part of the address field contains a network address; the second field contains the address of a specific computer.

A TCP/IP address is written in dotted-decimal notation, which is obtained by first dividing the 32-bit long address into four eight-bit long fields, and then converting each of the four fields into a decimal number and separating the fields with dots or periods.

Instead of using a numeric address, a symbolic name may be used to identify a computer in a TCP/IP network. Each computer in a TCP/IP network is assigned at last one name, which is resolved to a numeric address using either a translation file or an application known as the name server (which is part of the Domain Name System function of the Application layer).

TCP/IP Port

The notions of port and port number are extensions of the TCP/IP address. When the TCP/IP address has been used to deliver data to the desired computer in the network, the port number is used to identify the process for which the data are intended. By using ports and port numbers one computer can provide more than one service. CA XCOM Data Transport uses a predefined port number but it can be changed if it interferes with existing application services.

Conventions

The following is a list of standard conventions used in this manual.

Variable Input

Variable input is generally shown in lowercase letters and angle brackets, in the exact format that you are supposed to use (for example, <luname>). Variable input should not contain spaces. When a repeated letter such as nnnnnn is used, the number of letters represents the number of characters to be entered.

Ellipses

An ellipsis (...) is used to show that there are additional items not shown.

Programs, File Names, and Parameters

Programs, file names, and parameters are displayed in uppercase letters to distinguish them from the surrounding text (for example, the XCOMCNF file). This does not mean that they must be in uppercase when you use them in commands.

Commands and System Prompts

Command statements are displayed in uppercase letters in a different typeface to distinguish them from the surrounding text. For example:

```
SET <parameter_name>=<parameter_value>
```

Longer command statements are displayed in a table for easy reference.

Note: You must be aware of the case-sensitivity of the remote system when entering commands, programs, and parameters.

Documentation

The following guides are supplied with CA XCOM Data Transport for HP NonStop:

- *CA XCOM Data Transport for HP NonStop Product Guide*
- *CA XCOM Data Transport for HP NonStop Release Notes*

Chapter 2: Installing and Upgrading

This chapter contains system requirements, installation and upgrade procedures, as well as a list of CA XCOM Data Transport files. If you are installing the software for the first time or installing into a new volume.subvolume, you also need to configure the network and the software, as explained in later chapters.

This section contains the following topics:

System Requirements

CA XCOM Data Transport requires Guardian G06.29 or H06.06 or higher.

Network I/O Prerequisites

Both SNA and TCP/IP are layered architectures. Each layer is dependent on the services of the layer below it, but independent of the details of those lower layers. CA XCOM Data Transport sits at the highest layer—the end-user or application layer. It relies on the hardware vendor or a third party to provide the lower layers.

CA XCOM Data Transport has no hardware requirements of its own. The only hardware requirements are those of the lower layers of the network architecture.

- For SNA, CA XCOM Data Transport works with HP NonStop SNAX product line
- For TCP/IP, CA XCOM Data Transport works with HP NonStop's TCP/IP software

Network I/O

CA XCOM Data Transport supports two Network I/O subsystems. The method is selected by the APPC_TYPE parameter in the XCOMCNF file. The supported options are as follows:

APPC_TYPE Parameter Options	Network I/O Subsystems
SNAXAPPC	SNA stack provided by HP NonStop

APPC_TYPE Parameter Options	Network I/O Subsystems
TCPIP	TCP/IP stack provided by HP NonStop

OpenSSL Configuration

OpenSSL requires additional configuration parameters that must be specified in an SSL configuration file. For information about the SSL configuration file, see the chapter "Generating SSL Certificates."

Space Requirements

CA XCOM Data Transport requires approximately 18 MB of space on your system.

Access Requirements

To run CA XCOM Data Transport, you must have the following accesses:

- EXECUTE access to the program XCOM62
- READ access to the files EBCASC and ASCEBC (for translation purposes)

In addition, the program XCOM62 must have EXECUTE access to the APPC programs.

Distribution Media

CA XCOM Data Transport for HP NonStop is distributed on CD-ROM.

About Installing and Upgrading

To install or upgrade CA XCOM Data Transport, you need the following:

- The CA XCOM Data Transport distribution media
- A PC that can read the installation files from the CD and FTP them to the HP NonStop box
- The super.super logon

If you are installing for the first time, continue with the next section Install for the First Time. If you are upgrading, see Upgrade.

Install for the First Time

This section describes the first-time installation process, which should take a few minutes to complete.

Note: The prompt information you see on the screen when performing this procedure may be slightly different from the prompts presented here. Follow the prompts on the screen as needed.

To install CA XCOM Data Transport

1. Insert the CA XCOM Data Transport for HP NonStop CD into the CD-ROM drive into your PC.
2. Open a command prompt on your PC. At the command prompt, change the directory to the CD-ROM drive where the CA XCOM Data Transport for HP NonStop CD is accessible.
3. At the command prompt, type the following command and press Enter:

   ```
   ftp ip-address-of-your-NonStop-system
   ```

   Provide user ID and password when prompted. Within FTP, execute the following commands to change to the volume where you want to install the software, switch to binary mode, and cause the CA XCOM Data Transport for HP NonStop installation file to be transferred:

   For NonStop platforms:

   ```
   cd volume.subvolume
   binary
   put xctndm11
   ```

   For NonStop Integrity:

   ```
   cd volume.subvolume
   binary
   put xctndi11
   ```

4. Log on to HP NonStop and change to the volume to which you FTPed the installation file:

   ```
   volume volume.subvolume
   ```

5. Change the authority of the installation file:

   For NonStop platforms:

   ```
   FUP ALTER xctndm11, code 700
   ```

   For NonStop Integrity:

   ```
   FUP ALTER xctndi11, code 800
   ```

6. Logged on as super.super, enter the following command to install CA XCOM Data Transport:

   For NonStop platforms:

   ```
   RUN xctndm11/OUT $S.#CAXCOM/,*.*.*,VOL volume.subvolume,LISTALL,MYID
   ```

   For NonStop Integrity:

   ```
   RUN xctndi11/OUT $S.#CAXCOM/,*.*.*,VOL volume.subvolume,LISTALL,MYID
   ```

   For the variable, *volume.subvolume*, use the name of the volume (disk) and subvolume (directory) where the software is to be loaded. Choose any *volume.subvolume* other than $SYSTEM.SYSTEM. Files can be duplicated to $SYSTEM.SYSTEM later.

   Note: Though the CA XCOM Data Transport software can be placed in any subvolume, you should place it in a separate subvolume of its own. Installation messages are sent to $S.#CAXCOM.

7. Change to the volume to which you loaded the software:

   ```
   volume volume.subvolume
   ```

8. Create checkpoint file:

   ```
   FUP /IN MKCKPT/
   ```

9. Create history file:

   ```
   FUP /IN MKHIST/
   ```

10. (Optional) Create your CA XCOM Data Transport password file.

    ```
    FUP /IN MKPWF/
    ```

    Note: For more information, see the chapter "Security."

Note: After installation is complete, for configuration instructions, see the following chapters:

- "Configuring the Network"
- "Configuring CA XCOM Data Transport for HP NonStop"

Upgrade

This section contains the procedure for upgrading CA XCOM Data Transport. This is a simple procedure that takes a few minutes to complete.

Before You Upgrade

You need to take certain precautions before you upgrade.

Shut Down

Before upgrading or reinstalling, make sure all CA XCOM Data Transport jobs and processes are stopped, including XCOMDMN and XCOMPRNT processes. Coordinate with end users for the best time to perform this procedure, so that any CA XCOM Data Transport processes are not active.

Back Up Files Before Upgrading

Before your upgrade, back up any CA XCOM Data Transport files that may have been customized for your site. Examples include XCOMCNF and XCOMPWF.

Notes:

- The size and layout of the checkpoint files and the history files have changed.
- Additionally, if the information in the checkpoint files and the history files needs to be retained, back up these files **before** you upgrade. These files include CKPTFIL, CKPTALT, HISTALT, and XCOMHIST.

Start the Upgrade

Use the following command-line procedure to upgrade CA XCOM Data Transport.

Note: The prompt information you see on the screen when performing this procedure may be slightly different from the prompts presented here. Follow the prompts on the screen as needed.

To install CA XCOM Data Transport

1. Verify that CA XCOM Data Transport processes have been stopped.
2. Back up any existing files as needed (see Back Up Files Before Upgrading).
3. Insert the CA XCOM Data Transport for HP NonStop CD into the CD-ROM drive on your PC.
4. Open a command prompt on your PC. Change to the CD-ROM drive where the CA XCOM Data Transport for HP NonStop CD is accessible.
5. At the command line, type the following command and press Enter:

   ```
   ftp ip-address-of-your-Tandem-system
   ```

   Provide user ID and password as prompted.

6. Within FTP, execute the following commands to change to the volume where you want to install the software, switch to binary mode, and cause the CA XCOM Data Transport for HP NonStop installation file to be transferred.

   For NonStop platforms:

   ```
   cd volume-subvolume

   binary
   put xctndm11
   ```

   For NonStop Integrity:

   ```
   cd volume-subvolume

   binary
   put xctndi11
   ```

7. Log on to HP NonStop and change to the volume to which you FTPed the installation file.

   ```
   volume volume-subvolume
   ```

8. Change the authority of the installation file as follows:

   For NonStop platforms:

   ```
   FUP ALTER xctndm11, code 700
   ```

   For NonStop Integrity:

   ```
   FUP ALTER xctndi11, code 800
   ```

9. Logged on as super.super, enter the following command to install the software:

   For NonStop platforms:

   ```
   RUN xctndm11/OUT $S.#CAXCOM/,*.*.*,VOL volume.subvolume,LISTALL,MYID
   ```

   For NonStop Integrity:

   ```
   RUN xctndi11/OUT $S.#CAXCOM/,*.*.*,VOL volume.subvolume,LISTALL,MYID
   ```

   For the variable, *volume.subvolume,* use the name of the volume (disk) and subvolume (directory) where the software is to be loaded. Choose any *volume.subvolume* other than $SYSTEM.SYSTEM. Files can be duplicated to $SYSTEM.SYSTEM later.

   Note: Though the CA XCOM Data Transport software can be placed in any subvolume, you should place it in a separate subvolume of its own. Installation messages are sent to $S.#CAXCOM.

10. Restore any files that you backed up in step 2.
11. Change to the volume to which you loaded the software:

    ```
    volume volume.subvolume
    ```

12. If you are upgrading from a previous version of CA XCOM Data Transport, perform the following tasks:

    a. Recreate the checkpoint and history files, because the layout of the transfer record has been changed to accommodate new parameters.

    - To recreate the checkpoint files, enter the following commands at the command prompt:

      ```
      FUP PURGE CKPTFIL
      FUP PURGE CKPTALT
      FUP /IN MKCKPT/
      ```

    - To recreate the history files, enter the following commands at the command prompt:

      ```
      FUP PURGE HISTALT
      FUP PURGE XCOMHIST
      FUP /IN MKHIST/
      ```

    b. To use any existing CA XCOM Data Transport API applications, perform the following:

    - Ensure that the apiversion parameter is 3.
    - Recompile them with the new CA XCOM Data Transport API structures.
    - Bind them with the new APIO file.

13. (Optional) Create the CA XCOM Data Transport password file if desired and if password file XCOMPWF does not already exist. For more information, see the chapter "Security."

    ```
    FUP /IN MKPWF/
    ```

Program Files

The CA XCOM Data Transport software provides all of the program files listed below, including the following:

- TACL MACRO utilities for generating certificates
- Sample configuration files

Some of these sample configuration files can be used to perform transfers with CA XCOM Data Transport on specific platforms, but they are not necessary to run the program.

Use the following list to identify the program files:

AAREADME

Readme file describing the procedure and the sample files used to build EMS templates for CA XCOM Data Transport for Event Management Service (EMS).

APCLOG

A sample SNAX/APC log.

APIC

An example of a C program.

APICOB

An example of a COBOL program.

APIDDL

A template of the structure for the Data Dictionary Language file.

APIDEFT

An example of external declaration for the application programming interface.

APIEXS

An example of a TAL program.

APIO

An object file which provides a link with the application programming interface.

APPEND

A sample configuration file that specifies FILE_OPTION=APPEND.

AS400

A sample configuration file that specifies an AS/400 as the remote partner LU.

ASCEBC

A file used for ASCII to EBCDIC translation.

BINARY

A sample configuration file with CODE=BINARY.

CACONF

A file containing SSL parameters for creating a root certificate.

CLEANPEM

A utility for deleting certificates.

CLTCONG

A file containing SSL parameters for creating a client certificate.

CMDFILE

A sample obey file that performs transfers.

CMDFILE1

Another sample obey file that receives a file from a remote system.

CREATE

A sample configuration file that specifies FILE_OPTION=CREATE.

DEFINES

A file that provides the defines for the Event Management System.

EBCASC

A file used for EBCDIC to ASCII translation.

EMSCVIEW

A file that creates a process that reports all new CA XCOM Data Transport messages.

EMSFSDDL

A Data Dictionary Language file for the structure of definition for the Event Management Service.

EMSRVIEW

A file that reviews old CA XCOM Data Transport messages.

EMSTEST

A sample file with the defines necessary to set up a remote spooler.

EXTRADDL

A Data Dictionary Language file for the structure of definition for the Event Management Service.

GETST

A sample file transfer that uses substitution in the syntax.

IBMMVS

A sample configuration file used for linking an HP NonStop computer with an IBM mainframe.

IBMVSCNF

A sample configuration file used for linking an HP NonStop computer with an IBM mainframe.

IPCBLKC

A sample IPC program.

IPCRCNF

A sample IPC RECV_FILE configuration file.

IPCSCNF

A sample IPC SEND_FILE configuration file.

LIBCRYPTO

DDL Library for OpenSSL

LIBSSLSO

DDL Library for OpenSSL

LINKIAPI

Input file for the linker on an IA64 processor.

LINKIIPC

Input file for the linker on an IA64 processor.

LINKMAPI

Input file for the linker on a MIPS processor.

LINKMIPC

Input file for the linker on a MIPS processor.

LISTCA

A utility for listing root certificates.

LISTCLT

A utility for listing client certificates.

LISTSRV

A utility for listing server certificates.

MAKECA

A utility for creating root certificates.

MAKECLT

A utility for creating client certificates.

MAKEIAPI

A TACL MACRO for compiling the API on an IA64 processor.

MAKEIIPC

A TACL MACRO for compiling the API on an IA64 processor.

MAKEMAPI

A TACL MACRO for compiling the API on a MIPS processor.

MAKEMIPC

A TACL MACRO for compiling the IPC on a MIPS processor.

MAKESRV

A utility for creating server certificates.

MKCKPT

FUP input file used for recreating the checkpoint file.

MKHIST

FUP input file used for recreating the history file.

MKITPLS

Sample makefile to generate and install system templates for Event Management Service.

MKPWF

FUP input file used for recreating the password file.

OPENSSL

Command line tool for using the various cryptography functions of OpenSSL.

OTEST

A sample transfer file that used the HP NonStop PARAM function and an ADD DEFINE.

PATHCLD2

A sample configuration file for starting PATHWAY.

PATHCLD3

A sample configuration file for starting PATHWAY.

PATHCOLD

A sample configuration file for starting PATHWAY.

PATHCOOL

A sample configuration file for starting PATHWAY.

PATHLOG

A log file for remotely-initiated transfers.

PFILE2

A program that allows the user to add, edit, and delete user ID and password pairs from the password file XCOMPWF.

PUTST

A sample transfer file for a send transfer using substitution in the syntax.

PWCONF

A sample PATHWAY parameter file.

PWSTOP

A sample file to stop PATHWAY and SNAX.

README

A file that describes CA XCOM Data Transport r11.

SCANHIST

A program that allows the user to view the history file XCOMHIST.

SCFIN2

A sample SCF configuration file for SNAX/APC.

SCFIN3

A sample SCF configuration file for SNAX/APC.

SCFSDLC

A sample SCF configuration file for SDLC.

SCFTKNMF

A sample SCF configuration file for token-ring.

SRVCONF

A file containing SSL parameters for creating a server certificate.

STRATCNF

A sample configuration file for linking another PU 2.1 device.

STRATUS

A sample configuration file for linking another PU 2.1 device.

STRPWCLD

A sample of the file for cold starting the PATHWAY program.

STRTAPC

A sample file that starts SNAX/APC.

STRTPATH

A sample file used to startup the PATHWAY environment.

TESTALL

A sample transfer file used to print files on a Stratus computer.

TRCSTART

A file that starts the SCF trace.

TRCSTOP

A file that stops the SCF trace.

XAPIC

A file that supplies the API structure for C language programs.

XAPICOB

A file that supplies the API structure for COBOL.

XAPIFOR

A file that supplies the API structure for FORTRAN.

XAPIPAS

An application programming interface generation by the Data Dictionary Language file for programs in Pascal.

XAPITACL

A file that supplies the API structure for TACL.

XAPITAL

An application programming interface generated by the Data Dictionary Language file for programs in TAL.

XCM1ACF

A sample EMS filter for ACF.

XCM1C

A sample EMS filter for C.

XCM1COB

A sample EMS filter for COBOL.

XCM1DDLS

A sample EMS filter for DDLS.

XCM1EGEN

TAL object that is to be used by an application program.

XCM1EGES

TAL source file code for XCM1EGEN.

XCM1EMFO

A sample EMS test filter.

XCM1EMFS

A sample EMS test filter source.

XCM1INDX

Index file.

XCM1PROG

COBOL85 program to test the EGEN module.

XCM1PROS

COBOL85 source file code for XCM1PROG.

XCM1TACL

A sample EMS filter for TACL.

XCM1TAL

A sample EMS filter for TAL.

XCM1TEST

Sets up the DEFINES for EGEN and starts the XCM1PROG program.

XCM1UCOB

User-defined events number in a COBOL85 copylib format.

XCM1UDDL

User-defined events number in a DDL schema source file.

XCOM62

The CA XCOM Data Transport executable transaction program used to perform transfers.

XCOMCNF

A text file that contains the CA XCOM Data Transport parameter defaults.

XCOMDMN

Background process responsible for scheduling and automatic retries of failed transfers.

XCOMENCR

The CA XCOM Data Transport executable used to encrypt selected parameters in an existing configuration file.

XCOMIN

A transfer file that uses the OBEY and SET commands.

XCOMPRNT

The print process used by CA XCOM Data Transport to read spooled jobs.

XCOMQM

A file that lets you review and control scheduled transfers.

XCOMTPLS

A sample template source file for Event Management Service.

XCOMXFER

A sample transfer file that uses the HP NonStop PARAM function.

XQUEFCNF

A sample CA XCOM Data Transport configuration DEVICE file for sending files using the XCOMPRNT function.

XQUERCNF

A sample CA XCOM Data Transport configuration DEVICE file for sending reports using the XCOMPRNT function.

XCSSLCNF

A text file that contains the CA XCOM Data Transport SSL certificate parameter defaults.

ZSPLCOLD

A sample file that defines a spooler destination used by CA XCOM Data Transport to perform remote spooling functions.

ZSPLCONF

A sample file that defines a spooler destination used by CA XCOM Data Transport to perform remote spooling functions.

ZSPLWARM

A sample file that defines a spooler destination used by CA XCOM Data Transport to perform remote spooling functions.

Chapter 3: Configuring the Network

This chapter explains how to configure the network for CA XCOM Data Transport for HP NonStop (the second part of the installation and configuration process).

To provide LU 6.2 services, CA XCOM Data Transport for HP NonStop uses HP NonStop's SNAX/APC.

The following sections provide screen-by-screen instructions for CA XCOM Data Transport for HP NonStop to run on top of the SNAX/APC.

This section contains the following topics:

CA XCOM Data Transport and SNAX/APC

Tandem provides different products to support the various SNA layers. The only prerequisite for CA XCOM Data Transport is Tandem's SNAX/APC, which provides the APPC or LU6.2 layer.

Because CA XCOM Data Transport for HP NonStop does not interface directly with what runs below SNAX/APC, CA XCOM Data Transport for HP NonStop works with anything that SNAX/APC supports. At the physical unit layer, SNAX/APC can be used in the following configurations:

- **SNAX/XF** for PU 2.0 support for traditional hierarchical SNA networks, where VTAM on an IBM host is the master.
- **SNAX/CDF**, which provides host-like (PU 5) support for downstream PUs.
- **SNAX/APN**, which provides PU 2.1 LEN support for peer-to-peer APPN networks.

The lowest layer, the physical connection layer, is provided by TLAM for Token-ring and Ethernet LANs, X25AM for X.25, EnvoyACP for SDLC, and SNAXLINK for a channel connection.

Because SNAX supports PU 5, PU 2.0, and PU 2.1, CA XCOM Data Transport for HP NonStop can communicate with any other CA XCOM Data Transport system when using SNAX/APC.

- If you are connecting to a mainframe host, configure SNAX/APC as a PU 2 node.
- If you are connecting to a PC or a mini-computer, which can emulate only PU 2.0, configure SNAX as a PU 5 node to act as a host.
- For APPN networks, use SNAX/APN for LEN support.

Note: For more information about PU 2 and PU 5, see your SNA manual.

Plan Your SNAX/APC Configuration

This section explains the aspects of the SNAX/APC configuration that affect CA XCOM Data Transport for HP NonStop. The five steps below outline the general procedures and options for installing and configuring SNAX/APC on your system. Each step is discussed in more detail on the following pages.

SNAX/APC Configuration Overview

The following is a summary of the possible options you can use to configure your SNAX/APC configuration for CA XCOM Data Transport for HP NonStop.

1. Configure the SNAX Lines in SCF
   - Option 1: Use SCF with SNAX/XF
   - Option 2: Use SCF with SNAX/CDF
2. Create the empty SNAX/APC configuration files.
3. Run SNAX/APC as a PATHWAY server.
   - Option 1: Edit the supplied sample files.
   - Option 2: Create your own startup files.
4. Invoke the SNAX/APC configuration interface.
5. Test your SNAX/APC configuration.

For more information about using SNAX/XF, SNAX/CDF, SNAX/APC, or SNAX/APN, consult your HP NonStop manuals.

Note: If you have previously installed SNAX/APC, you can skip several of the early steps in the process. For example, you would not have to create the files mentioned in Step 3 because they already exist. However, you still need to modify the default values in XCOMCNF before you can use CA XCOM Data Transport for HP NonStop (see the chapter "Configuring CA XCOM Data Transport").

The SNAX/APC Configuration Worksheet

A worksheet of the key CA XCOM Data Transport for HP NonStop parameters is provided below. Confer with the remote system administrator when deciding the parameter values, and then use the worksheet as a reference when you fill in configuration values for SNAX/APC. Wherever you must use a specific value for a CA XCOM Data Transport for HP NonStop parameter, that value is filled in for you on the worksheet.

Note: The parameter descriptions appear in the tables beginning with the section, Line Parameters, later in this chapter.

Parameter	Configuration setting
APC-LU-NAME	
TP-NAMES	LU6SEND, LU6RECV, XCOMSEND, XCOMRECV
SNAX-FILE-NAME	

Parameter	Configuration setting
SERVER-CLASS-NAME	
SERVER-FILE-NAME	
RECEIVE-PACING	
PARTNER-LU-NAME	
MINOR APPLS (for TYPE=EXECUTE)	
PERIPHERAL-NODE	
SEND-PACING	
MODENAME	

The SNAX/APC Configuration Procedure

Configuring SNAX/APX requires the following steps:

- Step 1: Configure the SNAX lines in SCF
- Step 2: Create the empty SNAX/APC configuration files
- Step 3: Run SNAX/APC as a PATHWAY Server
- Step 4: Configure the SNAX/APC configuration interface
- Step 5: Start a session with a remote system

The following sections describe these steps.

Step 1: Configure the SNAX Lines in SCF

There are two options for performing this step:

- Option 1: Use SCF with SNAX/XF
- Option 2: Use SCF with SNAX/CDF

The following sections describe these options.

Option 1: Use SCF with SNAX/XF

Use the sample files SCFTKNMF and/or SCFSDLC, listed below, to configure the SNAX lines for your site. Modify the underlined parameters to agree with your own specifications. The parameter descriptions are listed in tables after the sample files.

Sample SCFTKNMF File

The following is a sample SCFTKNMF configuration file used for Token-ring connections.

```
ASSUME LINE $TKNA1


ALLOW ALL ERRORS

ASSUME LINE $RINGA

TRACE, STOP
ABORT LINE $RINGA, SUB LU
ABORT LINE $RINGA,SUB PU
ABORT LINE $RINGA

DELETE LINE $RINGA,SUB LU
DELETE LINE $RINGA,SUB PU
DELETE LINE $RINGA

ADD LINE $RINGA,LUOPMSG ON,MAXPUS 1,NOACQ OFF, &
RECSIZE 523,STATION SECONDARY,SWITCHED OFF,SWOPMSG ON, &
DIALTYPE INOUT,CHARACTERSET ASCII, SAPINFO (SAP %H04, MAXLS 2)

ADD PU $RINGA.#PU01,ADDRESS %HC1,MAXLUS 16,NOACQ OFF,PUIDBLK %H50, &
PUIDNUM %H0440,RECSIZE 521,REQMS OFF,ACTPU COLD,TYPE (13,2), &
ASSOCIATESUBDEV $TKNA1.#RNGAPU1, TRSSAP %H04, TRMAXOUT 15, TRMAXIN 15, &
TRRMTADDR 04400014000112

ADD LU $RINGA.#LU02,ADDRESS 2,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

ADD LU $RINGA.#LU03,ADDRESS 3,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

ADD LU $RINGA.#LU01,ADDRESS 1,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

ADD LU $RINGA.#LU04,ADDRESS 4,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)
```

```
ADD LU $RINGA.#LU05,ADDRESS 5,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

ADD LU $RINGA.#LU06,ADDRESS 6,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

STATUS LINE $TKNA1, DETAIL
DELAY 5

TRACE LINE $RINGA, TO SNAXTR, RECSIZE 1024, PAGES 1000
START LINE $RINGA
START LINE $RINGA,SUB PU
START LU $RINGA.#LU01
START LU $RINGA.#LU02
START LU $RINGA.#LU03
STATUS LINE $RINGA
STATUS LINE $RINGA,SUB PU
STATUS LINE $RINGA,SUB LU

STATUS LINE $TKNA1, DETAIL
DELAY 5

STATUS LINE $TKNA1, DETAIL
DELAY 5
```

Sample SCFSDLC File

The following is a sample SCFSDLC configuration file used for SDLC modem connections.

```
ALLOW ALL ERRORS

ASSUME LINE $SNA0

ABORT LINE $SNA0,SUB LU
ABORT LINE $SNA0,SUB PU
ABORT LINE $SNA0

DELETE LINE $SNA0,SUB LU
DELETE LINE $SNA0,SUB PU
DELETE LINE $SNA0

ADD LINE $SNA0,DUPLEX HALF,FLAGFILL OFF,LUOPMSG ON,MAXPUS 1,NOACQ OFF, &
POLLINT 0.30,RECSIZE 523,SPEED 9600,STATION SECONDARY,SWITCHED ON,SWOPMSG ON, &
DIALTYPE IN,AUTOACCEPT ON,CHARACTERSET ASCII

ADD PU $SNA0.#PU01,ADDRESS %HC1,MAXLUS 16,NOACQ OFF,PUIDBLK %H50, &
PUIDNUM %H0440,RECSIZE 521,REQMS OFF,ACTPU COLD,TYPE (13,2),WINDOW 7
```

```
ADD LU $SNA0.#LU02,ADDRESS 2,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

ADD LU $SNA0.#LU03,ADDRESS 3,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

ADD LU $SNA0.#LU01,ADDRESS 1,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

ADD LU $SNA0.#LU04,ADDRESS 4,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

ADD LU $SNA0.#LU05,ADDRESS 5,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)

ADD LU $SNA0.#LU06,ADDRESS 6,CHARACTERSET ASCII,NOACQ OFF,ACTLU WARM, &
PROTOCOL SNALU,PUNAME #PU01   ,RECSIZE 512,STATIC OFF,TYPE (14,0)


START LINE $SNA0
START LINE $SNA0,SUB PU
START LU $SNA0.#LU01
START LU $SNA0.#LU02
START LU $SNA0.#LU03
STATUS LINE $SNA0
STATUS LINE $SNA0,SUB PU
STATUS LINE $SNA0,SUB LU
```

The items underlined in the sample files must be configured as described in the tables below to generate a SNAX/APC line. For a full listing of the parameters for SNAX, see the Tandem manuals.

Line Parameters

The ADD LINE command defines the line parameters.

DUPLEX

Specifies whether the line is half or full duplex.

RECSIZE

Specifies the SDLC framesize in bytes. This includes the address and control bytes.

Range: 267 to 4096

STATION

Determines if you will communicate with a host or a PU 2 type node. If you are communicating with a host, define the station as SECONDARY. If you are communicating with a PC or mini-computer, define the station as PRIMARY.

PU Parameters

The ADD PU command defines the PU parameters used for configuring a SNAX/APC line.

ADDRESS

Specifies the SDLC secondary station address that identifies this PU in SDLC frames.

PUIDBLK

Specifies the ID block number for the PU.

PUIDNUM

Specifies a binary value that, together with the PUIDBLK, uniquely defines a PU in the network.

Note: You must specify PUIDBLK and PUIDNUM for PUs that will exchange SDLC XID commands and responses. For a PU for a secondary line, PUIDBLK and PUIDNUM can be given any values that are unique in the host's network, but they must match the IDBLK and IDNUM operands in the host's corresponding PU in the IBM ACF/VTAM gen.

RECSIZE

Specifies the maximum size in bytes for path information units (PIUs). The range is 16-4093. There is no default.

Note: You must add three extra bytes for the SDLC line RECSIZE.

TRRMTADDR

Identifies the mainframe front-end controller (for example, 3745, NCP) on the Token-ring network. The first two digits are the SAP, and the remainder is the Token-ring address. These values must be provided by the VTAM systems programmer or other individuals responsible for the SNA network at your site.

WINDOW

Specifies the maximum number of SDLC I-frames that can be sent to a remote station before receiving an acknowledgment. The range is one to seven. The default is the value of the window defined for the line associated with the PU.

LU Parameters

The ADD LU command defines the LU parameters used for configuring a SNAX/APC line.

ADDRESS

Specifies the LU's local address in the SNA environment. When communicating with a host, this is the LOCADDR defined for the LU in VTAM.

CHARACTERSET

Specifies whether the line is ASCII or EBCDIC. This value must be ASCII for SNAX/APC.

PROTOCOL

Specifies the protocol to be used. This value must be SNALU for SNAX/APC.

PUNAME

Specifies the name of the PU with which this LU is associated.

RECSIZE

Specifies the size in bytes of an RU.

Option 2: Use SCF with SNAX/CDF

To use SCF to configure SNAX/CDF, you must start the SNAX/CDF process as shown below:

```
CDFOBJ /NAME $CDF,NOWAIT,PRI 190/HOSTSA 14 NETNAME M14,MAXSUBA 255
```

NAME

Specifies the name of the SNAX/CDF process.

HOSTSA

Specifies the subarea address of the SNAX/CDF process.

NETNAME

Specifies the CDRM name of the SNAX/CDF process.

Note: For descriptions of these parameters, see your SNAX/CDF manual.

Now you can use SCF to configure SNAX/CDF. The following is a sample SCF OBEY file used to configure SNAX/CDF with a single ENVOYACP/XF line, a single PU, and a single APPL for use with SNAX/APC.

In the OBEY file's ADD APPL statement, the PROTOCOL must be SNALU and the CHARACTERSET must be ASCII for LU 6.2 applications.

```
ASSUME PROCESS $CDF
ADD ENTRY LOG1.XCOMMODE, TBLTYPE LOGMODE, TYPE %H00, FMPROF %H13, &
    TSPROF %H07,PRIPROT %HB0,SECPROT %HB0,COMPROT %H5081,RUSIZES %H8686,&
    PSSERVIC %H060200000000000000002C00, PSNDPAC %H05, SSNDPAC %H05, &
    SRCVPAC %H05
ADD PATH DESTSA20,ER0 (20,1), VR0 0
ADD LINE link1, TNDM $SDLC
ADD PU   pu1, LINE link1, TGN 1, SUBAREA 20, PUTYPE 4
ADD CDRM m20, SUBAREA 20
ADD CDRSC_SAXCOM,_CDRM m20
ADD APPL XCLU02,_OPENNAME #XCLU02, PROTOCOL SNALU, CHARACTERSET ASCII
START LINE link1
START PU PU1
START CDRM M20
START CDRSC CICS01
EXIT
```

For more information about the SNAX/CDF configuration, see the *SNAX/CDF Configuration and Control* manual.

Step 2: Create the Empty SNAX/APC Configuration Files

Before invoking the SNAX/APC interface, you must create the following empty files where the SNAX/APC configuration parameters will be stored:

- LUFILE
- PTRFILE
- TPNFILE
- MODFILE
- LUTPN
- LUPTR
- PTRMOD

To create these files

Enter the following command:

```
FUP /IN INFUP/
```

The INFUP file should be in the SNAX/APC system library.

Step 3: Run SNAX/APC as a PATHWAY Server

To define your SNAX/APC configuration, you have to run the SNAX/APC configuration program CONFIG. To run CONFIG, you must first run SNAX/APC as a PATHWAY server.

Configure for Remote Requests Using SNAX/APC

Because CA XCOM Data Transport for HP NonStop depends on the PATHWAY/DISPATCHER to start remotely initiated requests, CA XCOM Data Transport for HP NonStop requires SNAX/APC to be run in a PATHWAY environment. To process remote requests, you must be sure of the following:

- The SNAX/APC process must be a PATHWAY server.
- The following four servers must be defined:
  - LU6SEND
  - LU6RECV
  - XCOMSEND
  - XCOMRECV

Notes:

- LU6SEND and LU6RECV are used with Version 1 transfers.
- XCOMSEND and XCOMRECV are used with Version 2 transfers.

These server names are the names of the transaction programs for CA XCOM Data Transport for HP NonStop. For each of these servers, you must SET SERVER PROGRAM as follows:

volume.sub-vol.XCOM62

This command tells SNAX/APC to start a process that runs XCOM62 whenever an attach request is received for LU6SEND, LU6RECV, XCOMSEND, or XCOMRECV.

Option 1: Edit the Supplied Sample Files

CA XCOM Data Transport for HP NonStop includes the following sample files that you can modify to run SNAX/APC as a PATHWAY server:

- STRPWCLD
- PWCONF
- PATHCOLD

The STRPWCLD File

In the STRPWCLD file, modify direct references to *volume.subvolume* information and CPU specification to conform to your system. These references are underlined in the example below.

Note: Notice that the SNAX/APC process is given the name $SNAS, and the PATHWAY MONITOR process is named $SCI in these examples. You can change these names for your site, but be sure to be consistent throughout your configuration.

```
?TACL MACRO
== SNAX/APC RELEASE 3, XCOM VERSION 3  startup file

#FRAME
#PUSH #DEFAULTS #pmsg
pmsg on
volume $dsv.scisnax
#output  This is Strpwcold
#output  ~_~_~_~_Use Strtpath for a cool start of pathway
[#IF [#processexists $snas] |THEN|
    #output Stopping SNAXAPC server $SNAS
    STOP $SNAS
]
[#IF [#processexists $sci]  |THEN|
     #output Shutting down $SCI pathway
     pathcom $sci;abort term *;shutdown,wait
     DELAY 3 seconds
#output Purging the snaxapc and
     STOP $SCI
]
pmsg off pathway log files
fup purgedata (apclog, pathlog)
#output Cold starting $SCI pathway
pathmon/name $sci,cpu 1,nowait,out scisnax.pathlog,inspect saveabend/
pathcom/in scisnax.pathcld3/$sci
pathcom $sci;alter lu6send, in [#myterm], out [#myterm]
pathcom $sci;alter lu6recv, in [#myterm], out [#myterm]
pathcom $sci;alter xcomsend, in [#myterm], out [#myterm]
pathcom $sci;alter xcomrecv, in [#myterm], out [#myterm]
fup copy $dsv.scisnax.apclog,,share
#UNFRAME
```

The PWCONF File

The OBEY file PWCONF is supplied on the distribution media. The PATHCOLD file calls PWCONF to configure the PATHMON environment.

```
[ SNAX/APC  (T9096C20 07SEP90 AAR31C)
[ NOTE:
[ The following PATHWAY control file is provided as an
[ example only. You may need to modify this file before you
[ can use it to configure your SNAX/APC system.
[
[ The file has not been subjected to formal testing
[ procedures. The contents of this file will not be
[ supported by HP NonStop.
[
SET PATHMON BACKUPCPU 0
SET PATHWAY MAXTCPS 10
SET PATHWAY MAXTERMS 10
SET PATHWAY MAXPROGRAMS 12
SET PATHWAY MAXSERVERCLASSES 6
SET PATHWAY MAXSERVERPROCESSES 24
SET PATHWAY MAXSTARTUPS 10
SET PATHWAY MAXPATHCOMS 40
SET PATHWAY MAXASSIGNS 32
SET PATHWAY MAXPARAMS 32
SET PATHWAY MAXDEFINES 32
```

The PATHCOLD File

To provide a configuration for PATHWAY, the STRPWCLD file invokes the PATHCOLD file included in the SNAX/APC distribution subvolume. You must edit PATHCOLD to allow CA XCOM Data Transport for HP NonStop to run on top of SNAX/APC and to accommodate incoming remote requests from CA XCOM Data Transport on a remote system.

Change the underlined items in the example PATHCOLD file below to adjust the file for your site. The parameter settings are listed in the section after the sample file.

```
OBEY PWCONF                 [ Configure the Pathmon environment ]

START PATHWAY COLD!         [ Cold start the pathway ]

SET TCP PROGRAM $SYSTEM.SYSTEM.PATHTCP2
SET TCP CPUS 0:1
SET TCP MAXTERMS 15
SET TCP MAXSERVERCLASSES 010
SET TCP MAXSERVERPROCESSES 012
SET TCP MAXTERMDATA 08960
SET TCP MAXREPLY    02000
SET TCP NONSTOP 0
SET TCP STATS ON
SET TCP TCLPROG  $SYSTEM.SYSTEM.APCP
ADD TCP SNAXAPC-TCP

[Configure the SNAX/APC SERVER]
RESET SERVER
SET SERVER PARAM CONFIG  $DSV.SCISNAX.APCCFG
SET SERVER PARAM LOGFILE APCLOG
[ Stating a trace file automatically start tracing in release 3]
SET SERVER PARAM TRACEFILE trace4
SET SERVER PARAM BACKUPCPU -1
SET SERVER PARAM BACKUPDEBUG FALSE
SET SERVER PARAM COLLECTOR $0
SET SERVER PARAM DATAPAGES 4096
SET SERVER PARAM DEBUGONERROR FALSE
SET SERVER PARAM EMSSUPPRESS %H0300
[SET SERVER PARAM TRACEOPTION 8192]
SET SERVER PARAM TRACEOPTION -1
SET SERVER PARAM MAXINRUSIZE 512
SET SERVER PARAM MAXOUTRUSIZE 512
SET SERVER PARAM MAXAPPLIOSIZE  32000  [was 5120]
[SET SERVER PARAM ONESTEPREAD  1] [default is 2]
SET SERVER PARAM TRACEPAGES 256

SET SERVER LINKDEPTH 5   [ should agree with the number of terms ]
SET SERVER PROGRAM $SYSTEM.SYSTEM.APCOBJ
SET SERVER PROCESS $SNAS (ASSOCIATIVE ON)
SET SERVER NUMSTATIC 1
```

```
SET SERVER MAXSERVERS 1
SET SERVER CREATEDELAY 0 SECS
SET SERVER DELETEDELAY 1 MINS
SET SERVER CPUS 1:0
ADD SERVER SNAXAPCSVR

[Add CUSTOMER data TP server]
RESET SERVER
SET SERVER PROGRAM $DSV.XCOM307.XCOM62
[ SET SERVER PROCESS     ] [ Let pathway invent the process name ]
SET SERVER NUMSTATIC 0
SET SERVER MAXSERVERS 1
SET SERVER CREATEDELAY 0 SECS
SET SERVER DELETEDELAY 10 MINS
SET SERVER CPUS 1:0
SET SERVER IN $DIAL4
SET SERVER OUT $DIAL4
SET SERVER (PARAM XCOMCNF $DSV.XCOM307.XCOMCNF)
[SET SERVER DEBUG ON]          [uncomment to use Inspect on Xcom]
[Tell XCOM to use the default EMS collector]
SET SERVER (DEFINE =_EMS_COLLECTOR,       CLASS MAP, FILE $0)
[Tell XCOM to include message text in EMS messages]
SET SERVER (DEFINE =_EGEN_ADD_EVENT_TEXT, CLASS MAP, FILE $YES)
SET SERVER STARTUP "DISPATCH"          [ Required ]
ADD SERVER LU6SEND
RESET SERVER  PROCESS
ADD SERVER LU6RECV
RESET SERVER  PROCESS
ADD SERVER XCOMSEND
RESET SERVER  PROCESS
ADD SERVER XCOMRECV

[Configure the DISPATCHER]
SET TERM FILE $s.#displog
SET TERM INITIAL SNAXAPC-DISPATCHER
SET TERM TYPE CONVERSATIONAL
SET TERM TCP SNAXAPC-TCP
ADD TERM SNAXAPCSVR01 [First 10 chars are the SNAX/APC server name]
[ADD TERM SNAXAPCSVR02] [First 10 chars are the SNAX/APC server name]
[ADD TERM SNAXAPCSVR03] [First 10 chars are the SNAX/APC server name]
[ADD TERM SNAXAPCSVR04] [First 10 chars are the SNAX/APC server name]
[ADD TERM SNAXAPCSVR05] [First 10 chars are the SNAX/APC server name]

START SERVER SNAXAPCSVR
START TCP *
START TERM *
```

PATHCOLD General Parameters

Enter the parameter values listed below first. Then, if needed, edit the tracing and performance tuning parameters described in the subsequent tables.

ADD TERM

Specifies the SNAX/APC server name of a terminal designated to receive remotely scheduled transfers. The number of ADD TERMs indicates the number of terminals that can receive simultaneous active remote requests.

DELETEDELAY

Sets a time limit on how long the CA XCOM Data Transport for HP NonStop process will wait for another request.

MAXSERVERS

Indicates the maximum number of terminals or the maximum number of simultaneous active remotely scheduled transfers that CA XCOM Data Transport for HP NonStop can manage. Match this number to the number of ADD TERMs.

After the completion of each remotely initiated transfer, CA XCOM Data Transport for HP NonStop checks for additional remote requests. If the number of requests received exceeds the MAXSERVERS value, SNAX/APC queues the overflow.

NUMSTATIC

Equals the number in MAXSERVERS.

CPUS

Specifies your main and backup CPUs.

IN

Specifies an IN file. CA XCOM Data Transport for HP NonStop does not read the file, but the C library opens it. This is a required value.

OUT

Specifies an OUT file. This is a required value.

PARAM XCOMCNF

Defines your default CA XCOM Data Transport for HP NonStop configuration file. A PARAM XCOMCNF in your PATHWAY configuration tells the remotely started CA XCOM Data Transport which default configuration file to use.

For SNAX/APC transfers, set the parameter APPC_TYPE to SNAX/APC. Set the APPC_PROCESS_NAME to the SNAX/APC process name.

The following parameters are also set from XCOMCNF:

- IO_BUFFSIZE
- XBUFFSIZE
- PRI_ALLOC
- SET_ALLOC

DEFINE=_EMS_COLLECTOR

Sets the EMS collector for CA XCOM Data Transport, so that remotely initiated transfers will know about EMS.

STARTUP

Set this value to DISPATCH.

Performance Tuning Parameters

When CA XCOM Data Transport for HP NonStop is running, adjust the PATHCOLD parameters listed below to maximize the performance of CA XCOM Data Transport for HP NonStop. For more information, see the chapter "Configuring and Starting SNAX/APC" in the *SNAX/APC Manual*.

PARAM MAXINRUSIZE

Specifies the maximum RU size or the maximum number of bytes that CA XCOM Data Transport for HP NonStop can send or receive from the SNAX/APC.

PARAM MAXOUTRUSIZE

Specifies the maximum RU size or the maximum number of bytes in the outgoing SNA data field.

PARAM MAXAPPLIOSIZE

Specifies the largest buffer that CA XCOM Data Transport can send to SNAX/APC. The IO_BUFFSIZE parameter in the CA XCOM Data Transport for HP NonStop configuration file must not exceed this value.

Important! For SNA transfers to work, this parameter must be set to 32000.

LINKDEPTH

Agrees with the number of ADD TERMs defined in the PATHCOLD file.

Tracing Parameters

Set the following PATHCOLD parameters to use your trace options.

PARAM TRACE

Specifies whether the trace option is on.

Set this value to ON or OFF.

ASSIGN TRACE-FILE

Identifies the file to which the trace is written.

PARAM TRACEOPTION

Specifies the level of information traced:

- To see information passed across the line, select BIU.
- To trace all levels for basic problem solving, select -1.

Option 2: Create Your Own Startup Files

To create your own startup files for the PATHWAY environment

1. Define the CA XCOM Data Transport for HP NonStop transaction program names as servers. If you check the PATHCOLD file, you will see the ADD SERVER commands for LU6SEND, LU6RECV, XCOMSEND, and XCOMRECV.

   Notes:

   - Version 1 transfers use LU6SEND and LU6RECV.
   - Version 2 transfers use XCOMSEND and XCOMRECV.

2. Define the server program for these servers as XCOM62.
3. Set a SERVER PARAM for the default CA XCOM Data Transport for HP NonStop configuration file for remotely initiated requests.
4. Set the STARTUP message for both servers to DISPATCH.
5. Define your EMS_COLLECTOR for CA XCOM Data Transport.
6. If the name of your SNAX/APC process is not $SNAS (as shown in the sample files), make sure that all other references to this process are consistent with the name you choose.
7. Likewise, if you rename the PATHMON process (named $SCI in the sample files), make sure that all other references to this process are consistent.

Step 4: Configure the SNAX/APC Configuration Interface

See the Tandem SNAX/APN End Node Evaluation and Migration Guide (Web Version 2) for Configuration Interface.

You need to define the following:

- Configure the local TP names
- Configure your local APC LUs
- Configure your partner LUs
- Configure your mode entries

Sample SNAX/APC Configuration

CA XCOM Data Transport for HP NonStop has provided a sample SNAX/APC configuration. See below as a reference.

```
ASSUME PROCESS $SNAS
ALLOW ALL ERRORS
ABORT LU *, SUB ALL
DELETE LU *, SUB ALL

ADD LU LU440B03, SNANAME LU440B03, SNAXFILENAME $RINGA.#LU03,&
MAXSESSION 1, AUTOSTART YES
ADD PTNR-LU LU440B03.XCSDSDL, SNANAME XCSDSDL, PERIPHERAL-NODE 0,&
PARALLEL-SESSION-LU NO
ADD PTNR-MODE LU440B03.XCSDSDL.XCOMMODE, MODENAME XCOMMODE,&
DEFAULTMAXSESSION 1, DEFAULTMINCONWINNER 1, DEFAULTMINCONLOSER 0,&
MAXAUTOACT 0, RCVWINDOW 5, SENDWINDOW 5
ADD TPN LU440B03.LU6RECV, GENERALTPREADY YES , SESSIONCONTROL YES
ADD TPN LU440B03.LU6SEND, GENERALTPREADY YES , SESSIONCONTROL YES
ADD TPN LU440B03.XCOMRECV, GENERALTPREADY YES , SESSIONCONTROL YES
ADD TPN LU440B03.XCOMSEND, GENERALTPREADY YES , SESSIONCONTROL YES

ADD LU LU440B02, SNANAME LU440B02, SNAXFILENAME $RINGA.#LU02,&
MAXSESSION 1, AUTOSTART YES
ADD PTNR-LU LU440B02.XCOMMVS2, SNANAME XCOMMVS2, PERIPHERAL-NODE NO,&
PARALLEL-SESSION-LU NO
ADD PTNR-MODE LU440B02.XCOMMVS2.XCOMMODE, MODENAME XCOMMODE,&
DEFAULTMAXSESSION 1, DEFAULTMINCONWINNER 1, DEFAULTMINCONLOSER 0,&
MAXAUTOACT 0, RCVWINDOW 5, SENDWINDOW 5
ADD TPN LU440B02.LU6RECV, GENERALTPREADY YES , SESSIONCONTROL YES
ADD TPN LU440B02.LU6SEND, GENERALTPREADY YES , SESSIONCONTROL YES
ADD TPN LU440B02.XCOMRECV, GENERALTPREADY YES , SESSIONCONTROL YES
ADD TPN LU440B02.XCOMSEND, GENERALTPREADY YES , SESSIONCONTROL YES

START LU *, SUB ALL
```

Step 5: Start a Session with a Remote System

To start a session with the remote system

Enter the following command:

```
SCF /IN[set the File Name variable]/
```

This command runs the JCL.

When You Have Finished

After you complete the steps in this chapter, your environment is ready for use with CA XCOM Data Transport for HP NonStop. To begin Part 3 of the installation process, go to the chapter "Configuring CA XCOM Data Transport for HP NonStop."

CA XCOM Data Transport and TCP/IP

This section explains the tasks required to configure CA XCOM Data Transport to work with TCP/IP.

Locally Initiated Transfers

For TCP/IP, no configuration is necessary for locally initiated transfers.

Remotely Initiated Transfers

For remotely initiated transfers, you must do the following:

1. Define the CA XCOM Data Transport for HP NonStop listening port
2. Define the host name for HP NonStop
3. If necessary, direct the XCOM62 program to an *alternate* TCP/IP stack.

The following sections describe how to perform these tasks.

Define the CA XCOM Data Transport for HP NonStop Listening Port

Define the port to be used for CA XCOM Data Transport transfers to the HP NonStop LISTNER process. The HP NonStop LISTNER process is responsible for listening on predefined TCP/IP ports and starting the appropriate program to respond to incoming requests, like ftp, ping, echo, and finger.

To have the LISTNER start CA XCOM Data Transport for HP NonStop for remotely initiated requests (non-SSL and SSL), you must add lines to the PORTCONF file, then stop and restart the LISTNER. Be sure to keep the PORTCONF file in numerically ascending order. Following is what the PORTCONF file might look like:

```
#
# This file tells the listner program which ports to
# listen to, and what programs to run
# Telnet is supported directly by TELSERV, and does not
# use the listner's services.
# To run the listner use:
#       $system.ztcpip.listner / name.../ [config-file-name]
#  where config-file-name is this file.
#
ftp     $system.ztcpip.ftpserv
finger  $system.ztcpip.fingserv
7       $system.ztcpip.echoserv
# CA XCOM Data Transport port assignments: 8044 for non-SSL & 8045 for SSL
8044    volume.subvolume.xcom62
8045    volume.subvolume.xcom62 SSL
```

Start the HP NonStop TCP/IP listener as **super.super**. Otherwise, the ownership and access permission of the XCOM files (XCOMCNF, CKPTFIL, and so on) do not allow remote transfers. If the listener runs as super.oper, then the HP NonStop administrator must explicitly set the permissions for the XCOM files (AAAO).

Define the Host Name for HP NonStop

If the host name for HP NonStop is not defined correctly, the XCOM62 program abends.

To confirm that the host name is defined correctly

1. At the HP NonStop command prompt, enter the following:

   ```
   SCF
   ```

2. At the 1-> prompt, enter:

   ```
   INFO PROCESS $IP_process_name,detail
   ```

   $IP_process_name

   The name of the IP process.

3. If the IP process is not defined at all, define it by entering the following:

   ```
   ASSUME PROCESS $IP_process_name
   ```

4. If the IP process is defined incorrectly, correct the definition it by entering the following:

   ```
   ALTER PROCESS $IP_process_name,HOSTNAME hostname
   ```

   hostname

   The name by which the HP NonStop system space is known to the network. This name can be 49 or fewer alphanumeric characters.

Direct the XCOM62 Program to an Alternate TCP/IP Stack

To direct the XCOM62 program to an alternate TCP/IP stack

Set the following define statement:

```
ADD DEFINE=TCPIP^PROCESS^NAME, CLASS MAP, FILE\SYSTEM.$PROCESS
```

For More Information

For more information about using TCP/IP on HP NonStop , see the HP NonStop *TCP/IP Configuration and Management Manual*.

Chapter 4: Configuring the Software

The third part of the installation and configuration process requires configuring CA XCOM Data Transport for HP NonStop. To use the software most effectively, you need to understand how requests are made and how parameters are specified. This chapter describes the parameters and explains how to define their values.

This section contains the following topics:

About XCOMCNF

The software includes a default configuration file, XCOMCNF, that contains the CA XCOM Data Transport for HP NonStop file transfer parameters. You can specify any CA XCOM Data Transport parameter in the XCOMCNF file. The XCOMCNF defaults are used for every transfer unless you explicitly override them. The method you use to override the default values depends on which interface you use to enter the transfer request (for example, command line or API). Within XCOMCNF, the parameters are listed individually in the following format:

parameter_name=parameter_value

Note: You must be aware of the case-sensitivity of the remote system when entering commands, programs, and parameters.

parameter_name

Specifies the name of the parameter.

parameter_value

Indicates the option the user selects to perform a specific function.

The parameter values shown in this chapter are the provided default values, which you can edit using a text editor. These parameters provide default values for both locally and remotely initiated transfers. If a parameter is not defined in your default configuration file, CA XCOM Data Transport for HP NonStop takes the program default value.

Notes:

- For SNAX/APC, specify the default configuration file for remotely initiated transfers in your PATHCOLD file.
- For TCP/IP, the default configuration file should remain XCOMCNF.

Change the Configuration File Name

Because CA XCOM Data Transport for HP NonStop reads the default configuration file for each request, you can modify it for your environment. You can assign a file with a name other than XCOMCNF as your default configuration file by using the following command:

PARAM XCOMCNF *filename*

Change the Configuration File Contents

Edit the values in XCOMCNF to agree with your own setup. Several parameters are required to identify the target of the transfer request. The parameters required for transfers using SNAX/APC differ slightly from the parameters for TCP/IP.

Sample XCOMCNF File

A sample XCOMCNF is as follows:

```
#
# Local System Configuration Parameters
HISTORY_FILE=$CA3.XCOM300.XCOMHIST
RLOGFILE=$CA3.XCOM300
RLOG_SECURITY=NO
XDIR=$CA3.XCOM300
XLOGFILE=XCOMLOG
XLOG_FILE_TYPE=EDIT
XLUNAME=
#
# Remote Destination Configuration Parameters
#
REMOTE_SYSTEM=XCOMAPPL
APPC_TYPE=SNAXAPPC
# SNA/APPC Protocols Parameters
VERSION=2
APPC_OPEN_NAME=
APPC_PROCESS_NAME=$SNAS
CONV_SECURITY=NO
XMODE=XCOMMODE
# TCP/IP Protocols Parameters
PORT=8044
SOCK_DELAY=YES
SOCK_RCV_BUF_SIZE=0
SOCK_SEND_BUF_SIZE=0
TCP_TIMEOUT=60
TXPI_BUF_SIZE=32760
TXPI_SEND_CHECK_FREQ=10
TCP_RECEIVE_TIMEOUT=
#
# Transfer Parameters
#
# General Transfer Parameters
LOCAL_FILE=
REQUEST_NO=
```

```
TRANSFER_ID=
TRANSFER_USER_DATA=
SYSTEM_USER_DATA=
# Record Handling Parameters
CARRIAGE_FLAG=YES
NULLFILL=NO
# EBCDIC/ASCII Translation Parameters
CODE_FLAG=EBCDIC
ASCEBC=$CA3.XCOM300.ASCEBC
EBCASC=$CA3.XCOM300.EBCASC
CODETABL=
# File Parameters
REMOTE_FILE=
FILE_OPTION=CREATE
CREATEDELETE=
# HP NonStop Disk File Creation Parameters
DEALLOC_EXTENTS=NO
FILE_CODE=0
GUARDIAN_FILE_TYPE=NONE
MAXEXTENTS=
# IBM Mainframe File Creation Parameters
RECORD_FORMAT=VB
LRECL=
BLKSIZE=0
UNIT=
VOLUME=
ALLOC_UNIT=B
PRI_ALLOC=2
SEC_ALLOC=4
DIR_ALLOC=0
# Report Parameters
# LRECL for remote printing must be specified as 00133.
CARRIAGE_CONTROL_CHARACTERS=OTHER
CHARS=
CLASS=
COPIES=1
DESTINATION=
DISPOSITION=DELETE
FCB=
FORM=
HOLD_FLAG=NO
REPORT_TITLE=
SPOOL_COLLECTOR=$S.#XCOMJOB
SPOOL_FLAG=YES
# Job Parameters
# The DISPOSITION parameter controls whether the temporary EDIT file is deleted
# after the JOB is submitted.
SPOOL_JOBNUMBER=YES
#
```

```
# Performance Options
#
CACHEBUF=NO
COMPRESS=YES
COMPRESS_PDS=
# For SNAX, the IO_BUFFSIZE value must be less than or equal to the SNAX/APC
# MAXAPPLIOSIZE parameter.
IO_BUFFSIZE=31744
PACK=NO
SIO=NO
XBUFFSIZE=4096
#
# Special Feature Parameters
#
# IPC Interface Parameters
IPC_NO_REMOTE=NO
IPC_PNAME=
IPC_FNAME=
# Checkpoint/Restart Parameters
RETRIES=0
RETRY_TIME=60
RESTART_FLAG=NO
RESTART_SUPPORTED=YES
CHECKPOINT_COUNT=0000
CHECKPOINT_FILE=$CA3.XCOM300.CKPTFIL
# Testing and Tracing
XTRACE=0
RTRACEFILE=$CA3.XCOM300
# Store-and-Forward Parameters
# REMOTE_SYSTEM specifies the name of the remote system as defined in the
# Destination Table on the mainframe.
XIDEST=
# Notification Parameters
LOCAL_NOTIFY=
NOTIFY_NAME=
NOTIFYR=NONE
LCLNTFYL=
RMTNTFYL=
# Security Parameters
PASSWORD=
PASSWORD_FILE=NONE
USERID=
TRUSTED=
DOMAIN=
# Scheduling Transfer Parameters using the XCOMDMN
START_DATE=
START_TIME=
EURO_DATE=NO
```

```
# Tape Parameterss
TAPE=
EXPDATE_FLAG=D
TAPE_LABEL=
DEN=
EXPDT=
RETPD=
UNITCT=
VOLCT=
VOLSQ=
LABELNUM=
TAPEDISP=
# SMS Parameterss
STORCLAS=
DATACLAS=
MGMTCLAS=
DSNTYPE=
SECLABEL=
# CA XCOM Gateway
GATEWAYGUID=
# OpenSSL Parameters
SECURE_SOCKET=NO
XCOM_SHOW_CIPHER=NO
XCOM_CONFIG_SSL=
```

When You Have Finished

After modifying the CA XCOM Data Transport for HP NonStop configuration parameters to your user site specifications, you are ready to begin transferring files. For more information, see the chapter "The Batch and Command Line Interface."

CA XCOM Data Transport Parameters

CA XCOM Data Transport transfers require many parameters to establish a link between CA XCOM Data Transport, the APPC, and a remote system. A default configuration file, XCOMCNF, contains default values for these parameters.

Because the configuration parameters are changed only when switching remote systems and/or changing the APPC, we suggest that you edit their values in the XCOMCNF file first. If you do not want to edit the XCOMCNF settings, you can also enter each parameter explicitly on the command line or specify it in one or more additional configuration files.

Although all of the CA XCOM Data Transport parameters reside in a single file, it is helpful to consider them in groups according to their function.

The following categories of parameters are in this chapter:

Local System Configuration Parameters

The Local System Configuration Parameters define how CA XCOM Data Transport operates on the local system.

Remote Destination Configuration Parameters

The Remote Destination Configuration Parameters define the link between CA XCOM Data Transport, the APPC, and the remote partner. These values usually remain the same unless you are switching partners or APPCs.

Transfer Parameters

The Transfer Parameters define the particular CA XCOM Data Transport transfer that you want to make. They include the following:

- General Transfer Parameters
- Record Handling Parameters
- File Parameters
- Tandem Disk File Creation Parameters
- IBM Mainframe File Creation Parameters
- Report Parameters
- Job Parameters

Performance Options Parameters

The Performance Options Parameters allow you to define how CA XCOM Data Transport is to run on your system.

Special Feature Parameters

The Special Feature Parameters define specific CA XCOM Data Transport functions for your system. They include the following:

- Checkpoint/Restart Parameters
- Testing and Tracing Parameters
- Store and Forward Parameters
- Notification Parameters
- Security Parameters

OpenSSL Parameters

The OpenSSL Parameters define specific CA XCOM Data Transport SSL functions for your system. They include the following:

- Secure transfer
- Show ciphers

Local System Configuration Parameters

This section describes the parameters that are used for local transfers:

HISTORY_FILE

Specifies the name of the history file to which the history records are written. Use the following format:

vol.subvol.filename

Range: Up to 27 characters.

Default: xcomhist

RLOG_SECURITY

When using SNAX in a remotely initiating scenario, CA XCOM Data Transport for HP NonStop acts as a server handling multiple, serial transfer requests from the SNAX Dispatcher. This makes the system more efficient, because it is not necessary to create a new CA XCOM Data Transport for HP NonStop process for every transfer. For each transfer, CA XCOM Data Transport for HP NonStop logs in as the user requested in the transfer.

Remote log and trace files are named by the system date (for example, RT950418). When a new file is created during the server's existence, it uses the user ID of the latest successful transfer, and thus this file inherits the ownership and security attributes of that user ID. It is not possible for CA XCOM Data Transport for HP NonStop to revert back to the original user ID used at process creation time without the password.

If you do not want the new log file to be created with the last transfer's user ID, set the parameter RLOG_SECURITY to Y. If you need to create a new log file, CA XCOM Data Transport for HP NonStop stops, the SNAX Dispatcher creates a new CA XCOM Data Transport process, and the trace and log files are created with the correct user ID.

Range: Y or N

Default: N

XDIR

Specifies the default volume and subvolume for all files that are read and written by CA XCOM Data Transport for HP NonStop except the XCOMCNF, XCOMHIST, XCOMPWF, and CKPTFIL files.

Note: If a transfer is initiated remotely and XDIR is not specified, CA XCOM Data Transport uses the default volume of the user ID that the remote system sends.

Range: Up to 256 characters.

Default: None

XLOG_FILE_TYPE

Controls which Guardian file type are created.

Because multiple processes write simultaneously to the entry sequence files, the process name is provided in addition to the time stamp. If it is an unnamed process, the PID is used.

If the log/trace file already exists, it ignores this parameter.

Range: EDIT and ENTRYSEQ

Default: EDIT

XLOGFILE

For locally initiated CA XCOM Data Transport transfers only.

Provides a file name for the log file for locally initiated transfers.

Range: Up to 250 characters.

Default: XCOMLOG

XLUNAME

Identifies the local LU name to use for SNAX.

Range: Up to eight characters.

Default: None

Remote Destination Configuration Parameters

If you have several different remote destinations, it is a good idea to create separate configuration files containing only the destination information for each remote system. For example, if you are sending to a PC and to a mainframe using SNAX/APC, create a configuration file for each system type.

The configuration file for the PC might look like this:

```
APPC_PROCESS_NAME==$SNA
APPC_TYPE=SNAXAPPC
REMOTE_SYSTEM=LUPC
XMODE=XCOMMODE
XLUNAME=LUSNAX
```

The configuration file for the z/OS mainframe might look like this:

```
APPC_PROCESS_NAME=$SNAS
APPC_TYPE=SNAXAPPC
REMOTE_SYSTEM=XCOMAPPL
XLUNAME=LUSNAX
XMODE=XCOMMODE
```

As a result, when you enter a request to transfer a file to either system, you only have to specify the configuration file containing the parameters for that system. For example, if you want to send a file to the PC and you have already created a configuration file named PCCNF, your command looks like this:

```
RUN XCOM62 PUT FILE1 AS FILE.PC, PCCNF
```

If you want to send a file to the z/OS mainframe and you have already created a configuration file named MVSCNF, your command looks like this:

```
RUN XCOM62 PUT FILE1 AS FILE.MVS, MVSCNF
```

General Remote Destination Configuration Parameters

The remote destination configuration parameters define the link between CA XCOM Data Transport for HP NonStop, the APPC, and the remote partner. These values usually remain the same unless you are switching partners or APPCs.

This section describes the remote destination configuration parameters that are not specific to a protocol. To identify some of the values necessary to configure these parameters, use the Configuration Worksheet provided in the chapter "Configuring the Network."

APPC_TYPE

Required.

This parameter indicates your APPC configuration or protocol type.

Range: SNAXAPPC or TCPIP

Default: SNAXAPPC

REMOTE_SYSTEM

For SNA

The LU name

For TCP/IP

The remote system's IP address, host name, or domain name

For indirect transfers

(That is, for store-and-forward transfers to CA XCOM Data Transport for z/OS or z/VSE that have another final destination), the REMOTE_SYSTEM name is the name of the final destination as defined in the CA XCOM Data Transport destination table on the mainframe.

Range: Up to 128 characters

Default: XCOMAPPL

VERSION

Locally initiated transfers only.

Indicates whether the request is a Version 1 or Version 2 transfer.

1

Version 1

2

Version 2

Default: 2

Transferring Files Using SNA/APPC Protocols

This section contains information about performing file transfers from the command prompt using SNA/APPC protocols. Use this information as examples when performing transfers that use SNA/APPC protocols.

Using SNA/APPC Protocols

Your computer and the remote CA XCOM Data Transport system must be configured for the appropriate SNA/APPC for you to use SNA/APPC protocols with CA XCOM Data Transport for HP NonStop.

The choice of protocol to use is indicated by the APPC-TYPE parameter. This can be specified at the command line or in the configuration file (XCOMCNF), depending upon your installation's needs. If the protocol is not specified at the command line, the defaults specified in the configuration file are used.

For more information about your platform's SNA/APPC configuration, see your SNA/APPC vendor documentation.

Parameters for SNA/APPC Transfer Protocols

The following parameters are used with SNA/APPC transfers.

APPC_PROCESS_NAME

The name of the process used by CA XCOM Data Transport. This name must agree with the process name specified in the SNAX/APC configuration.

Example:

If you used the supplied PATHCOLD file to start SNAX/APC, the APPC_PROCESS_NAME is $SNAS.

Range: Up to 16 characters

Default: None

CONV_SECURITY

Applies to locally initiated SNAX transfers.

Specifies whether the user ID/password pair is to be sent in the SNA ATTACH request. On the mainframe, CONV_SECURITY is controlled by the ACCSEC parameter in the CA XCOM Data Transport Destination Table (XCOMCNTL).

YES

Sends the user ID/password pair in the ATTACH request.

NO

User ID/password pair is not sent in the ATTACH request.

Default: NO

XMODE

Identifies the SNA LOGMODE for SNAX.

Range: Up to eight characters

Default: XCOMMODE

Transfer Files Using TCP/IP Protocols

This section contains information about performing file transfers from the command prompt using TCP/IP protocols. Using IP addresses, host names, or domain names when performing transfers that use TCP/IP protocols are also discussed.

Using IP Addresses and Names

Your system and the remote CA XCOM Data Transport system must be configured for TCP/IP before you can use TCP/IP protocols with CA XCOM Data Transport for HP NonStop. If your computer is not configured for TCP/IP, check with your network administrator for more information.

Before performing a file transfer, you must know the IP address, and either the host name or the domain name of the remote system. Check with the network administrator of the remote system for these values.

The formats of the IP address, host name, and domain name are as follows:

IP address

A unique number for a particular computer that is used to identify it on the TCP/IP network. IP addresses are in the dotted decimal notation format.

Example: 123.123.12.11

Host name

The host name of a particular computer.

Example: goodsys

Domain name

The Domain Name Service (DNS) name. Identifies the computer's group in the DNS hierarchy. The host name and the domain name make up the fully qualified domain name of the computer.

Example: goodsys.goodsite.com

Goodsys is the name of the computer, which is in the goodsite.com domain.

Specifying the Remote System

When using TCP/IP, the remote system can be specified in different ways. For example, you can use the following forms:

By host name

REMOTE_SYSTEM=goodsys

By fully qualified domain name

REMOTE_SYSTEM=goodsys.goodsite.com

By IP address

REMOTE_SYSTEM=123.123.12.11

The IP address is the most efficient method when specifying a remote system location.

TCP/IP Name Resolution

If you use a host name or domain name, any symbolic name that can be mapped to an IP address can be used to resolve that name to an IP address. However, your system must be set up to resolve the name to an IP address. If you are relying on name resolution to resolve names to IP addresses, this capability must be installed and configured on your system.

There are many ways to implement name resolution; typical ways include the Domain Name Service (DNS), and the use of host files. For more information about your system's use of names, check with your network administrator.

TCP/IP Protocol Parameters

If you want to use TCP/IP protocols for a CA XCOM Data Transport for HP NonStop transfer, you must specify the port for the remote system and indicate that you wish to use TCP/IP protocols.

The port is specified by the PORT parameter. The default value in the XCOMCNF file should be valid for most remote hosts. If you need to change the port value of the local system, see the appropriate CA XCOM Data Transport installation guide for your platform.

The choice of protocol to use is indicated by the APPC_TYPE parameter. This can be specified at the command line or in the XCOMCNF file, depending upon your installation's needs. If the protocol is not specified at the command line, the defaults specified in the configuration file are used.

In addition, CA XCOM Data Transport for HP NonStop uses other parameters for TCP/IP functionality. These parameters are of interest to system administrators for tuning and performance considerations. They are described next, and are also listed in the appendix "Configuration File Parameters."

PORT

The number of the TCP/IP port on the remote CA XCOM Data Transport server. Used for TCP/IP transfers only.

Range: 1 to 65535

Default: 8044

SOCK_DELAY

This parameter refers to the Nagle algorithm for send coalescing. By default, small sends may be delayed, but this should have no impact for normal CA XCOM Data Transport for HP NonStop record sizes. Used for TCP/IP transfers only.

Note: Socket options affect the way CA XCOM Data Transport for HP NonStop uses TCP/IP stack implementation.

YES

Small sends may be delayed. (Does not turn on the socket option TCP_NODELAY, and does not disable the Nagle algorithm)

NO

All sends are immediate. (Disables the Nagle algorithm)

Default: YES

SOCK_RCV_BUF_SIZE

The default TCP/IP Socket option is SO_RCVBUF. This parameter can be used to specify the buffer size for receives. Use zero for the default size provided by the socket implementation. The value for SOCK_RCV_BUF_SIZE can be smaller than the value for TXPI_BUF_SIZE. Used for TCP/IP transfers only.

Note: Socket options affect the way CA XCOM Data Transport for HP NonStop uses the TCP/IP stack implementation.

Range: 0 to 32760

Default: 0

SOCK_SEND_BUF_SIZE

The default TCP/IP Socket option is SO_SNDBUF. This parameter can be used to specify the buffer size for sends. Use zero for the default size provided by the socket implementation. The value for SOCK_SEND_BUF_SIZE can be smaller than the value for TXPI_BUF_SIZE. Used for TCP/IP transfers only.

Note: Socket options affect the way CA XCOM Data Transport for HP NonStop uses the TCP/IP stack implementation.

Range: 0 to 32760

Default: 0

TCP_RECEIVE_TIMEOUT

Specifies the period of time that CA XCOM Data Transport will wait for a TCP socket call to complete.

You can instruct CA XCOM Data Transport to wait no longer than *nn* seconds for a TCP socket call to complete. On HP NonStop, CA XCOM Data Transport uses nowait TCP socket calls, followed by the AWAITIOX system call. This parameter is used by AWAITIOX to determine how long to wait for completion before assuming failure. Zero means an unlimited wait.

Range: 0 to 999

Default: 60 (one minute)

TXPI_BUF_SIZE

TXPI_BUF_SIZE specifies the internal buffer size for sends and receives. The default size allows multiple CA XCOM Data Transport for HP NonStop records to be received in a single socket call. With this default, CA XCOM Data Transport for HP NonStop tries to receive multiple records in a single socket call, if the CA XCOM Data Transport for HP NonStop record size is less than 32K. Used for TCP/IP transfers only.

Range: 0 to 65536

Default: 32760

TXPI_SEND_CHECK_FREQ

This parameter indicates how frequently CA XCOM Data Transport for HP NonStop checks to see if incoming error information is available when sending data. For example, if the value is 5, a check is made every fifth time that data is sent, to determine if data is available for receiving. Larger values give better performance. Smaller values minimize the sending of data after the partner reports an error. Used for TCP/IP transfers only.

Range: 1 to 9999

Default: 10

OpenSSL Parameters

The following parameters are used with OpenSSL transfers.

XCOM_CONFIG_SSL

Specifies the name of the SSL configuration file. Use the following format:

vol.subvol.filename

Range: Up to 27 characters.

Default: XCSSLCNF

Note: The value for XDIR will be used if specified.

XCOM_SHOW_CIPHER

Specifies whether to display encryption algorithms in the CA XCOM Data Transport queue detailed information, which is used for transfers.

NO

Do not display encryption algorithms in the queue detail information.

YES

Display encryption algorithms in the queue detail information.

Default: NO

SECURE_SOCKET

Specifies whether to use an OpenSSL socket or non-OpenSSL socket for transfers.

YES

Performs a secure transfer. The transfer uses an OpenSSL socket and must to connect to an SSL listener on the remote partner.

NO

Performs a non-secure transfer. The transfer uses a non-OpenSSL socket.

Default: NO

Transfer Parameters

The Transfer parameters define the particular CA XCOM Data Transport for HP NonStop transfer that you want to make. They include:

- File parameters
- Disk file creation parameters
- File transfer parameters
- Report parameters
- Job parameters

The following sections describe these groups of parameters that are used to customize CA XCOM Data Transport for HP NonStop file and report transfers.

General Transfer Parameters

Use the following parameters to define file, report, or job transfers.

LOCAL_FILE

Identifies the name of the file on the local system. Tandem file naming conventions apply.

Important! For scheduled transfers, you must specify the full path name.

Range: Up to 256 characters

Default: None

TRANSFER_ID

Specifies the non-unique user-assigned identifier for each transfer.

Range: Up to 10 characters

Default: None

TRANSFER_USER_DATA

Indicates any user-specified information for each transfer that can be passed to the remote system. This information is written in the history record.

Range: Up to 10 characters

Default: None

Record Handling Parameters

Use the following parameters for record handling.

CARRIAGE_FLAG

Controls the treatment of text files

If CARRIAGE_FLAG=YES and CODE_FLAG is ASCII or EBCDIC, new line characters are added to incoming records and removed from outgoing records.

The Tandem file system, like mainframe and AS/400 file systems, does not use record separators. When transferring text files with a system that does use record separators (UNIX or PC), make sure that the other system adds them when receiving files and removes them when sending files.

Range: YES or NO

Default: YES

NULLFILL

Indicates whether CA XCOM Data Transport for HP NonStop is to fill the end of outgoing EBCDIC text records with null characters.

N

Do not use null characters at the end of the record.

Y

Use null characters at the end of the record.

Default: NO

EBCDIC/ASCII Translation Parameters

Tandem uses the ASCII code set. When transferring text files with a mainframe or an AS/400, the Tandem system is responsible for data translation.

This feature is controlled by the following parameters.

CODETABL

Applies to Windows, Linux, and UNIX partners only.

Specifies the prefix to the custom character conversion file names on Windows, Linux, or UNIX that will be used by the transfer.

Range: Zero to three alphanumeric characters

Default: None

CODE_FLAG

Identifies the type of data being transferred.

Important! CA XCOM Data Transport translates every byte in the file. If you have mixed character and binary data, the file will be corrupted if you specify EBCDIC.

EBCDIC

Translation is required when sending a file.

ASCII

Translation is required when receiving a file.

BINARY

No translation is required. Specify BINARY if a binary file such as an executable file is being transferred.

Default: ASCII

ASCEBC

Specifies which file to use for ASCII to EBCDIC conversion.

If a file name is entered, CA XCOM Data Transport uses that file for the translation. If there is no value entered, or if CA XCOM Data Transport cannot find the file, CA XCOM Data Transport uses the default settings (the same as the deliverable tables).

The parameter format is as follows:

ASCEBC=*vol.subvol.filename*

Note: If you enter commands from different subvolumes, you must specify the full *vol.subvol.filename* for this parameter. Make sure that all users of this file have the correct access.

Range:

- Up to 8 characters for *filename*
- Up to 26 characters for *vol.subvol.filename*

Defaults:

- For *filename*: ascebc
- For *vol.subvol.filename*: None

EBCASC

Specifies the file to use for EBCDIC to ASCII conversion.

If a file name is entered, then CA XCOM Data Transport uses that file for the translation. If there is no value entered, or if CA XCOM Data Transport cannot find the file, CA XCOM Data Transport uses the default settings (the same as the deliverable tables).

The parameter format is as follows:

EBCASC=*vol.subvol.filename*

Note: If you enter commands from different subvolumes, you must specify the full *vol.subvol.filename* for this parameter. Make sure that all of the users of this file have the correct access.

Range:

- Up to 8 characters for *filename*
- Up to 26 characters for *vol.subvol.filename*

Defaults:

- For *filename*: ebcasc
- For *vol.subvol.filename*: None

File Parameters

Use the following parameters to define a file transfer:

REMOTE_FILE

Indicates the name of the file on the remote system.

If you are creating the file, make sure your designated file name is consistent with the file naming conventions of the remote system. The remote system (not the local CA XCOM Data Transport system) determines whether the file name is valid.

Range: Up to 256 characters

Default: None

FILE_OPTION

Indicates how the transferred data is to be processed by the receiving system.

CREATE

Creates a new file on the receiving system.

APPEND

Appends this data to an existing file on the receiving system.

REPLACE

Replaces the contents of an existing file on the receiving system. On HP NonStop, if the file does not exist, it is created automatically.

Default: CREATE

HP NonStop Disk File Creation Parameters

This section contains information that you need to create, name, and transfer files for CA XCOM Data Transport for HP NonStop.

HP NonStop File Naming Conventions

HP NonStop file IDs consist of the following parts:

system identifier

The system identifier uniquely identifies an HP NonStop system on an EXPAND network.

volume identifier

The volume identifier uniquely identifies a physical disk drive.

sub-volume identifier

Sub-volumes are created when a new name is used when creating a file. They are similar to subdirectories on other systems. Valid names are one to eight alphanumeric characters, but the first character must be a letter.

file name

File names can be one to eight alphanumeric characters, but the first character must be a letter.

Note: File IDs are not case sensitive.

The components of a file ID are separated by periods. For example:

```
\SYS1.$DSV.QAXCOM.BIGFILE
```

Types of Files Supported

Format 1 and 2 are static attributes of a file, which get established during file creation. The Format 2 files were introduced with the D46 release and differ from Format 1 files as follows:

- Larger partitions than the current 2GB less 1MB Format 1 file partitions
- Larger primary keys and alternate-key records for relative and entry sequenced files.

When you create an Enscribe file, you can specify the maximum amount of physical disk space to be allocated for that file in the form of extents. An extent is a contiguous block of disk space that can range in size from a single page (2048 bytes) to 65,535 pages (134,215,680 bytes) for Format 1 files or to 536,870,912 pages for Format 2 files.

File Formats 1 and 2 are both supported by CA XCOM Data Transport. The user is not required to specify the format, however, if it is left unspecified, the system chooses Format 1 unless Format 2 is required, which happens when an extent size is over 65,535 pages.

CA XCOM Data Transport for HP NonStop supports the following file types through ENSCRIBE, HP NonStop's disk file architecture:

Structured

- Relative
- Entry Sequence
- Key Sequence

Unstructured

- EDIT

The following sections describe these structured and unstructured file types.

Structured Files

Taken as a group, Relative, Entry Sequence, and Key Sequence are known as structured files. Structured files are record oriented. These file structures are very much like VSAM on the mainframe.

Relative

Relative files are ordered by relative record number. The space allocated for each record is specified when the file is created. Records in these files can be deleted and re-added in place. Relative files contain fixed length physical records, but each physical record contains a length attribute and a variable length data portion (logical record).

Entry Sequence

Entry Sequence files are sequential files. Records are stored in the order they are entered. These records are variable in length and cannot be added or deleted. They are accessed by their record address.

Key Sequence

Key Sequence files contain variable length records. CA XCOM Data Transport for HP NonStop cannot dynamically create a Key Sequence file. When receiving data to a Key Sequence file on Tandem, the file must already exist. CA XCOM Data Transport for HP NonStop supports only FILE_OPTION=REPLACE or FILE_OPTION=ADD for Key Sequence files. If this is a new file, it must be predefined using FUP, the Tandem File Utility Program.

Unstructured Files

An unstructured disk file is essentially a large byte array, much like the stream-oriented file system on UNIX. The organization of an unstructured file (the lengths and locations of records within the file) is entirely the responsibility of the application.

EDIT

EDIT files are unstructured files whose organization is determined by the HP NonStop editor. On other systems, an EDIT file might be referred to as a TEXT file. An edit file should contain only printable characters.

For more information about ENSCRIBE and unstructured files, see the *ENSCRIBE Programmer's Guide*.

File Codes

The use of a file can be denoted by the file code. File codes 100 to 999 are reserved. File code 100 denotes an object file or executable code. File code 101 identifies an EDIT file. We use file code 1001 for our checkpoint and history files. For locally initiated create and receive transfers, specify the file code using the FILE_CODE parameter.

Format 2 Files

The following parameters must be carefully configured both for the type of transfer being performed and in relation to each other. When transfers are sent or received, you must use the SPACE parameters to dictate whether a Format 1 or Format 2 file is to be created.

- PRI_ALLOC
- SEC_ALLOC
- DIR_ALLOC

PRI_ALLOC

The primary extent size for creating local and remote files.

Range: 1 to 32767

Default: 2

SEC_ALLOC

The secondary extent size for creating local and remote files.

Range: 1 to 32767

Default: 4

DIR_ALLOC

Specifies the number of directory blocks to allocate when creating a PDS data set on a remote z/OS system. This corresponds to MAXEXTENTS on HP NonStop.

Range: 0 to 32767

Default: 0

File Type Specification

You can use the following parameters to specify the file type, depending on the type of send or receive request and on the remote system:

- GUARDIAN_FILE_TYPE
- RECORD_FORMAT

The following sections describe these variations.

Locally Initiated Send Requests

When sending a file from HP NonStop to another system, CA XCOM Data Transport for HP NonStop determines what type of file it is and branches to the appropriate code. Similarly, when receiving a file to HP NonStop, if the file already exists, CA XCOM Data Transport for HP NonStop determines what type of file it is and branches to the appropriate code. When receiving a file to a new data set on HP NonStop, you must specify what kind of file it is.

Locally Initiated Receive Requests

For locally initiated receive requests, the file type must be specified by the GUARDIAN_FILE_TYPE parameter. Choose from the following values:

- EDIT
- UNSTRUCTURED
- ENTRYSEQ
- RELATIVE

Remotely Initiated Send Requests

For remotely initiated transfers, or if GUARDIAN_FILE_TYPE=NONE is coded in the XCOMCNF file or GUARDIAN_FILE_TYPE is not present, then the value of RECORD_FORMAT determines the type of disk file.

VB

EDIT

F

Relative

FB

Entry Sequence

U

Unstructured

Other Systems

On other systems, the RECORD_FORMAT parameter has other names, as follows:

RECORD_FORMAT

HP NonStop/UNIX

RECFM

Mainframe

RECFMT

AS/400

RECFM

PC

-format

Stratus

Disk File Creation Parameters

When CA XCOM Data Transport for HP NonStop creates an HP NonStop disk file, it uses the following parameters to determine the type of ENSCRIBE file to create. These parameters must be carefully configured both for the type of transfer being performed and in relation to each other. When transfers are initiated from an IBM mainframe system, you must use the SPACE parameters, which are described in the section SPACE Parameters.

GUARDIAN_FILE_TYPE

Indicates the type of Enscribe file to create.

EDIT

Creates an EDIT file.

ENTRYSEQ

Creates an Entry Sequenced file.

RELATIVE

Creates a relative file.

UNSTRUCTURED

Creates an unstructured file.

NONE

The type of file is determined by the RECORD_FORMAT value.

Default: NONE

DEALLOC_EXTENTS

Returns any unused extents to the system when a file is closed.

If set to YES, a CONTROL 21.0 is executed to deallocate all unused extents past the end of file.

Range: YES or NO

Default: NO

FILE_CODE

Specifies the Guardian Enscribe file code when creating a file on the local HP NonStop system.

Range: 0 to 9999

Default: 0

MAXEXTENTS

Sets a limit lower than the default for this file type:

- For EDIT files, the default is 900.
- For all other file types, the default is 256.

Notes:

- For a remotely initiated transfer, the DIR_ALLOC parameter is used.
- For a z/OS initiated transfer, this is the last item in the SPACE parameter.

IBM Mainframe File and Tape Parameters

Use these parameters to provide additional information to an IBM mainframe system when you create a file on those systems (FILE_OPTION=CREATE).

SMS Information

DATACLAS

Specifies the name of the data class to use when allocating a new SMS-managed data set.

Note: This parameter applies only to mainframe SMS data sets.

Range: One to eight characters

Default: None

DSNTYPE

Specifies the data set definition.

Note: This parameter applies only to mainframe SMS data sets.

LIBRARY

Defines a PDSE.

PDS

Defines a partitioned data set.

Note: These values are IBM standards for SMS processing.

Range: One to eight characters

Default: None

MGMTCLAS

Specifies the name of the management class to use when allocating a new SMS-managed data set.

Note: This parameter applies only to mainframe SMS data sets.

Range: One to eight characters

Default: None

STORCLAS

Specifies the name of the storage class for a new SMS-managed data set.

Note: This parameter applies only to mainframe SMS data sets.

Range: One to eight characters

Default: None

Data Set Actions

COMPRESS_PDS

Applies to z/OS only.

COMPRESS_PDS is the parameter that causes the actual PDS compression to happen. If your CA XCOM Data Transport z/OS administrator has enabled the programmatic PDS compression feature in a CA XCOM Data Transport region, you can use the COMPRESS_PDS option to control if and when output PDS data sets get compressed as part of the transfer.

Note: COMPRESS_PDS applies only to PDS data sets that will be, or have been, opened for output as the target of a CA XCOM Data Transport transfer.

NONE

Suppresses the compression of an output PDS data set as part of a CA XCOM Data Transport transfer.

BEFORE

Causes an output PDS data set to be compressed before the transfer of user data begins.

AFTER

Causes an output PDS data set to be compressed after the transfer of user data has completed.

BOTH

Causes an output PDS data set to be compressed both before and after the transfer of user data.

Default: NONE

CREATEDELETE

Applies to z/OS only.

CREATEDELETE specifies whether an existing z/OS data set should be deleted and a new data set allocated at the start of a FILE_OPTION=CREATE transfer.

YES

If FILE_OPTION=CREATE and the data set exists, then the z/OS data set is deleted and a new data set is allocated at the start of the transfer.

NO

If FILE_OPTION=CREATE and the z/OS data set exists, then the transfer fails with a catalog/file error.

Default: NO

Notes:

- Specifying CREATEDELETE=YES causes the attributes of the existing data set to be lost; the new data set is allocated with the attributes specified in the transfer.
- CREATEDELETE applies only if the target data set is a sequential data set or an entire PDS/PDSE. CREATEDELETE is ignored for other types of data sets (such as PDS members, PDSE members, VSAM, and USS files).
- CREATEDELETE does not apply to relative GDGs unless the data set is specified using the fully qualified G*xxxx*V*xx* name.
- The use of CREATEDELETE=YES must be allowed by your site's CA XCOM Data Transport administrator for z/OS through the default table (XCOMDFLT) or destination member (XCOMCNTL).

Tape Information

DEN

Specifies the density to be used in creating a tape on the remote system. Valid values are the same as those for the DEN parameter in JCL.

Range: 1 to 4

Default: None

EXPDT

Specifies an expiration date for the tape data set in terms of a two-digit designation for the year and a three-digit designation for the day of the year.

Example:

In the expiration date 11021, 11 is the year (namely, 2011) and 021 is the 21st day of that year, when the tape data set expires.

Format: *yyddd*

Default: None

Note: EXPDT and RETPD are mutually exclusive; specify one or the other.

LABELNUM

Indicates the sequence number of the data set on the tape.

Sequence number (0001 to 9999)

This value identifies the sequence number of a data set on tape.

Example:

LABELNUM=2

This specification refers to the second data set on the tape.

Default: 0001

RETPD

Specifies the number of days (1 to 9999) that the tape data set being created is to be retained.

Range: 1 to 9999

Default: None

Note: RETPD and EXPDT are mutually exclusive; specify one or the other.

TAPE

Indicates to the remote system whether the volume is a tape volume or a disk file.

YES

Indicates a tape volume and that mounts are allowed when performing dynamic allocation.

NO

Indicates that the transfer is to a disk file.

Default: None

TAPE_LABEL

Indicates the type of label associated with a tape data set. The following table lists the valid values for this parameter.

Processing type (AL, AUL, BLP, LTM, NL, NSL, SL, SUL)

Represents the type of processing to be applied to data sets on tape.

Note: CA XCOM Data Transport for z/OS supports only standard label tapes.

Example:

LABEL=BLP

The type of processing to be applied to this data set is BLP.

Default: AL

TAPEDISP

Specifies the disposition value for MVS tape data sets.

1

New

2

Old

3

Mod

Default: 1

UNITCT

Specifies the number of units to be allocated on the remote system. This is a tape parameter and is used when the partner is an IBM mainframe.

Range: 1 to 20

Default: None

VOLCT

Specifies the maximum number of volumes to be used in processing a multi-volume output tape data set on the remote system.

Range: 1 to 255

Default: None

VOLSQ

Specifies the sequence number of the first volume of a multi-volume remote data set to be used.

Range: 1 to 255

Default: None

DCB Information

Use these parameters to provide DCB information.

RECORD_FORMAT

Specifies the record format for the file being created. This corresponds to the JCL RECFM subparameter.

F (Fixed Unblocked)

All records have the same length.

FB (Fixed Blocked)

Fixed record length with multiple records per block.

VB (Variable Blocked)

Variable record length with multiple records per block.

U (Undefined)

Undefined record length.

Default: VB

LRECL

Specifies the actual or maximum length in bytes of a logical record. This corresponds to the JCL LRECL subparameter.

For a variable blocked format

LRECL should equal the maximum record length.

For a fixed or fixed blocked format

LRECL should equal the constant record length.

Range: Up to five characters

Default: 0, except in the following cases:

- If GUARDIAN_FILE_TYPE=EDIT or UNSTRUCTURED, the default is 239, or 243 for variable blocked.
- If GUARDIAN_FILE_TYPE=RELATIVE or ENTRYSEQ, the default is taken from the record length parameter in the transferred file.

BLKSIZE

Specifies the physical block size of a file. The range depends on record length.

For a variable record format

BLKSIZE = LRECL + 4

For a fixed or fixed blocked record format

BLKSIZE = a multiple of LRECL

For an undefined record format

BLKSIZE > largest record length

Note: If you create a structured file on the HP NonStop system, it must be a valid HP NonStop block size. CA XCOM Data Transport computes an appropriate value.

Range: Up to five characters

Default: 4096

Device Selection

Use these parameters to provide additional information for device selection.

UNIT

IBM Mainframe File Creation parameter

Specifies the unit on which to create the file. Ignored for files created on the Tandem.

Range: Up to six characters

Default: None

VOLUME

IBM Mainframe File Creation parameter

Specifies the volume on which to create the file.

Range: Up to six characters

Default: None

SPACE Parameters

Use these parameters when you create a file on, or when transfers are initiated from, an IBM mainframe system. The syntax is as follows:

```
SPACE=(ALLOC_UNIT=value,(PRI_ALLOC=value,SEC_ALLOC=value,DIR_ALLOC=value))
```

ALLOC_UNIT

Used only when creating mainframe files.

Specifies the size of the allocation unit if the remote is an IBM mainframe. The actual byte count of each type will vary, depending on the storage device.

B

Blocks

C

Cylinders

T

Tracks

Default: B

Note: If you have questions about allocation units, consult your System Administrator.

PRI_ALLOC

The primary extent size for creating local and remote files.

Range: 1 to 32767

Default: 2

SEC_ALLOC

The secondary extent size for creating local and remote files.

Range: 1 to 3567

Default: 4

DIR_ALLOC

Specifies the number of directory blocks to allocate when creating a PDS data set on a remote z/OS system. This corresponds to MAXEXTENTS on HP NonStop.

Range: 0 to 32767

Default: 0

Report Parameters

Use the following parameters to define report properties.

CARRIAGE_CONTROL_CHARACTERS

Indicates the type of carriage control characters that are used in the print job.

ASA carriage control characters are as follows:

Blank

Space 1 line

0

Space 2 lines

-

Space 3 lines

+

Suppress space

1

Skip to line 1 on new page

Valid options are as follows:

ASA

ASA control codes in column 1.

- When sending a disk file from Tandem to print on a remote system, specify ASA if the disk file has ASA carriage control characters in column one. Otherwise, specify OTHER. You should choose IBM only if you previously sent a file from a mainframe with machine carriage control characters to a Tandem disk file, and now want to send it back to a mainframe for printing.
- When the XQUE feature is specified, this parameter controls whether ASA codes are generated when CA XCOM Data Transport sends the file from the Tandem spooler to the remote system.
- When a report is sent to a Tandem system, the Tandem interprets the ASA characters if the remote partner specified ASA. Tandem does not support the IBM machine code carriage control characters.

IBM

IBM Machine Characters (valid for a z/OS remote system only).

OTHER

No carriage control codes.

Default: OTHER

CHARS

Reports only.

Specifies the font for reports sent to a z/OS system. For more information, see your z/OS manual.

Range: Up to four characters

Default: None

CLASS

Reports only.

Indicates the print class for the print job.

If the remote system is a z/OS system, then CLASS designates the JES SYSOUT class. In this case, to print the report through SYSOUT=B, enter B.

Note: If printing on HP NonStop, this parameter is ignored.

Range: One character

Default: None

COPIES

Reports only.

Indicates the number of copies to be printed when a remote system sends a report to CA XCOM Data Transport for HP NonStop.

Range: Up to three characters

Default: 1

DESTINATION

Reports only.

Indicates the print job's destination on the remote system. If no destination is specified, the remote system sends the job to the system's default printer.

For report printing on Tandem systems, the remote system should specify the destination as follows:

$<*collector*>.#<*location*>

If no COLLECTOR is specified, then SPOOL_COLLECTOR is used.

Range: Up to 21 characters

Default: None

DISPOSITION

Reports only.

Indicates the disposition of the printed file after printing is completed. Whether this parameter is meaningful is system dependent.

DELETE

Delete the file after it is printed.

KEEP

Do not delete the file.

HOLD

Hold after printing.

Default: DELETE

FCB

Indicates the forms control block (FCB) JCL parameter when sending the report file to a z/OS mainframe. It defines print density, lines per page, and so on.

Note: FCB is ignored for report printing on HP NonStop systems.

Range: Up to four characters

Default: None

FORM

Specifies which forms the printed output should use.

When a remote system sends a report to CA XCOM Data Transport for HP NonStop, the FORM parameter must identify a valid Tandem Spooler form.

Valid names can contain letters, digits, and blanks only. Invalid characters result in the transfer being failed with a SPOOLSTART error 4097.

Because CA XCOM Data Transport places the print job in the remote system's print queue, the print control functions will depend on the remote system. Before sending the report, you must verify that the form you are requesting is available at the remote site.

Note: When sending a report to an OpenVMS system, leave FORM blank unless you are certain that the value is a valid form type. OpenVMS interprets a blank to mean that no special form is being requested.

Range: Up to 10 characters

Default: The default form for the remote printer

HOLD_FLAG

Reports only.

Indicates the transferred report file's HOLD status on the remote system.

Valid options are as follows:

YES

Hold the report (spooled on a z/OS system).

NO

Prepare the report for immediate printing.

Default: NO

REPORT_TITLE

Reports only.

Provides the report name to be printed on the job separator when a remote system sends a report to CA XCOM Data Transport for HP NonStop.

Valid names can contain letters, digits, and blanks only. Invalid characters result in the transfer being failed with a SPOOLSTART error 4097.

The title is interpreted depending on the type of remote (receiving) system, as follows:

i5/OS (AS/400)

CPF assumes this to be the printer file name.

z/OS

A non-blank value generates a separator (banner) page.

OpenVMS

This title will be printed with the report.

UNIX

This field will be passed to the lp spooler as a title field.

Other systems

This field is generally used only as a descriptive comment and is not printed as part of the report.

Range: Up to 21 characters

Default: None

SPOOL_COLLECTOR

Indicates the default location for reports received from a remote system. For jobs received from a remote system, the output of the job will be written to this spool collector.

The format is as follows:

$<collector>.#<location>

collector

The name of a spooler process. The standard Tandem default spooler collector name is $S.

location

Similar to a job name on other systems.

When CA XCOM Data Transport for HP NonStop receives a job file, a new TACL process is created to execute the file. The output file for this job is the file defined for SPOOL_COLLECTOR.

Range: Up to 25 characters

Default: $S.#XCOMJOB

SPOOL_FLAG

System-dependent flag.

Indicates to the remote system whether it should spool the report received. HP NonStop sends all the reports that it receives to the spooler.

YES

Spool the report received from the local system.

NO

Do not spool the report.

Default: YES

Job Parameters

When you send a job to the HP NonStop system, the file is stored in an EDIT file with a file name of TEMPZ*nnn*, where *nnn* is a three-digit number. The job is then submitted for execution using the Guardian NEWPROCESS procedure. If the job is submitted successfully, the transfer is considered to be successful. CA XCOM Data Transport for HP NonStop does not report back on the success or failure of the job itself, only on whether or not it is successfully submitted.

The following parameters are used when CA XCOM Data Transport for HP NonStop receives a job.

DISPOSITION

Reports only.

Indicates the disposition of the printed file after printing is completed. Whether this parameter is meaningful is system dependent.

DELETE

Delete the file after it is printed.

KEEP

Do not delete the file.

HOLD

Hold after printing.

Default: DELETE

JOB_TIME_OUT

Specifies the period of time that CA XCOM Data Transport is to wait for a send job to complete. You can instruct CA XCOM Data Transport to wait no longer than *nnnnn* seconds for a send job to complete.

Zero means no waiting.

Range: 0 to 86400

Default: 0

Note: If a remote send job takes longer than the number of seconds specified for JOB_TIME_OUT, the transfer terminates with an error 40 (the operation timed out).

With such an error 40, even if the parameter DISPOSITION is set to DELETE, the temporary file holding the remote TACL commands TEMP*xxxx* does not get deleted and the job is left running. It is up to the user to investigate, stop the job, purge the TEMP*xxxx* file, and take corrective action (increase the JOB_TIME_OUT value or change the processing) so that the error 40 does not occur again.

SPOOL_JOBNUMBER

When a remote system performs a SEND JOB to HP NonStop, this parameter controls the value of the jobid parameter passed to the Guardian NEWPROCESS procedure call.

YES

The job ID of the CA XCOM Data Transport process is passed to NEWPROCESS. The CA XCOM Data Transport process is the ancestor of the newly created TACL process. If the number of jobs for each spooler queue is limited by the Tandem administrator, and you reach the limit, then the job fails with error 14.

NO

No job ID is supplied to the NEWPROCESS call. If the CA XCOM Data Transport process is not part of a job, neither is the new process. If the CA XCOM Data Transport process is part of a job, the new process is part of the same job.

ZERO

A value of zero is passed to the NEWPROCESS call. The new process will not be part of any job.

The number of jobs for each spooler queue can be limited by the HP NonStop administrator.

- If SPOOL_JOBNUMBER=YES, CA XCOM Data Transport honors this limit.
- If SPOOL_JOBNUMBER=NO, CA XCOM Data Transport ignores this limit.

Range: YES, NO, and ZERO

Default: YES

Performance Options

Use the following parameters to define the performance of CA XCOM Data Transport for HP NonStop.

CACHEBUF

Writes records to cache instead of directly to disk. If the cache buffer becomes full, the records are written to disk.

Cache buffering is a standard Guardian option. Because cache buffering is set on each disk drive, performance varies from disk to disk. For more information, see the PUP manual.

YES

Turns cache buffering on.

NO

No cache buffering.

Default: NO

COMPRESS

Indicates the transmission type. Compressing data may decrease transmission time.

NO

Do not compress the data transmission buffers. This option is used when the CPU resource is more of a constraint than network bandwidth.

YES

The original CA XCOM Data Transport compression method for reducing strings of multiple blanks and nulls. Provided for backward compatibility.

RLE

Run length encoding of any repeating characters. This is the least CPU intensive of the compression methods.

COMPACT

Run length encoding of any repeating characters, plus a two-byte compaction algorithm suitable for uppercase English text.

LCOMPACT

Run length encoding of any repeating characters, plus a two-byte compaction algorithm suitable for mixed case English text.

LZSMALL | LZMEDIUM | LZLARGE

Lempel-Ziv derivatives for small, medium, and large memory models. They achieve the greatest reduction in the data transmitted, but consume the most CPU time.

HUFFMAN

This option selects a basic Huffman encoding technique. This technique tends to provide greater compression than RLE, but not as much as the Lempel-Ziv 77 derivatives. Huffman uses more CPU than RLE, but generally uses less than the Lempel-Ziv 77 derivatives.

LZRW3

The LZRW3 algorithm is a general purpose compression algorithm that runs quickly and gives reasonable compression. The algorithm is a member of the Lempel-Ziv family of algorithms, and bases its compression on the presence of repeated substrings in the data.

Next to RLE, this is the least expensive compression option in terms of CPU utilization. It will not reduce the data transmitted by as much as ZLIB, LZSMALL, LZMEDIUM, and LZLARGE, but it usually consumes far less CPU time. With some data, this method will use less CPU time than HUFFMAN, while providing greater compression.

ZLIB*n*

Greater compression than LZRW3 but less than LZSMALL, LZMEDIUM, and LZLARGE. The *n* value can be 1 through 9. ZLIB is a Lempel-Ziv 77 derivative. This technique tends to provide greater compression than LZRW3, but somewhat less than LZSMALL, LZMEDIUM, and LZLARGE. CPU utilization tends to be much greater than LZRW3 and RLE, but somewhat less than LZSMALL, LZMEDIUM, and LZLARGE.

Default: YES

Compression usage guidelines are as follows:

- When sending text files to an IBM AS/400, COMPRESS should be set to YES to overcome the problem of zero-length lines. Compression guarantees that all lines will have at least one character to satisfy the LU 6.2 read on the receiving end.
- Most of the current CA XCOM Data Transport releases support all compressions. Check the product documentation for each platform for details.
- COMPRESS=YES is provided for backward compatibility with older releases of CA XCOM Data Transport. For any current release, COMPRESS=RLE is a better choice.
- COMPRESS=RLE is inexpensive in terms of CPU utilization and, for text files, is recommended over COMPRESS=NONE.
- COMPRESS=LZRW3 is the least expensive of the advanced compression methods in terms of CPU utilization, and should be tried first. If you are CPU bound, you may get better wall clock time using RLE, COMPACT, COMPACTL, or LZRW3, rather than HUFFMAN, ZLIB, LZSMALL, LZMEDIUM, or LZLARGE.
- COMPACT and COMPACTL add a byte compaction scheme on top of RLE. They may compress text files slightly better than RLE, without adding much in terms of CPU utilization.
- Use Huffman, or any of the Lempel-Ziv 77 derivatives, only if you are using packing (for example, PACK=BIG) or have an LRECL of 500 or greater. The output buffer that CA XCOM Data Transport tries to compress must be at least 500 bytes long for these compression methods to be effective. In some cases compression is disabled if the output buffer is less than 500 bytes. If you do not use packing and your LRECL is less than 500 bytes, then you should use RLE, COMPACT, or LCOMPACT.

  For maximum benefit from Huffman or any of the Lempel-Ziv 77 derivatives, you should code PACK=BIG and IO_BUFFSIZE=32000 in the XCOMCNF file.
- The slower the communications link, the more important compression is to data transfer speeds. Conversely, the faster the communications link, the less important compression will be. If you have a fast communications link, you may see little difference in wall clock time between transfers using COMPRESS=RLE versus transfers using COMPRESS=LZLARGE.

- When examining CPU utilization, you have to compare both the time it takes to compress the data and the time it takes to expand the data. Particularly with the Lempel-Ziv derivatives, compression tends to take more CPU time than decompression. This consideration could be important if one of the transfer partners is more CPU constrained than the other.
- The choice of which compression method to use depends on several factors:
  - Communications line speed
  - CPU utilization constraints
  - Nature of the repetitiveness of the data

For your routine production jobs, you are encouraged to experiment with the various compression methods to see which one will provide the best compromise between network I/O and CPU utilization for your data in your environment.

IO_BUFFSIZE

Used with SNAX/APC.

Specifies buffer size when HP NonStop sends a file to another system. IO_BUFFSIZE lets you maximize throughput and eliminate excessive overhead in interprocess communication between CA XCOM Data Transport for HP NonStop and SNAX/APC.

Notes:

- When Big Packing is used, this parameter controls how large the pack buffers will be (similar to the MAXPACK parameter in the CA XCOM Data Transport for z/OS destination table (XCOMCNTL)).
- When Big Packing is not used, this parameter controls the size of the buffers passed from the CA XCOM Data Transport for HP NonStop process to the SNAX process.
- For SNAX, the IO_BUFFSIZE value must be less than or equal to the SNAX/APC MAXAPPLIOSIZE parameter value. In general, the IO_BUFFSIZE should be higher for higher speed lines.

Range: From 4136 to 32000, inclusive

Default: 31744

PACK

Packs up to 31KB of data into a buffer before transmission to a remote system. The receiving system is responsible for unpacking the record(s).

NO

Does not use record packing. Each logical record is sent out individually.

YES

Uses packing feature with a 2KB buffer.

Notes:

- On Windows, Linux, and UNIX platforms, this is specified by CARRIAGE_FLAG=MPACK.
- On z/OS, this is specified by the combination of PACK=LENGTH and MAXPACK=2048.

BIG

Uses packing feature with up to a 31KB buffer.

Notes:

- On Windows, Linux, and UNIX platforms, this is specified by CARRIAGE_FLAG=XPACK.
- On z/OS, this is specified by the combination of PACK=LENGTH and MAXPACK with a value greater than 2048.
- When HP NonStop is sending the file, the packing block size is determined by the IO_BUFFSIZE parameter.
- When HP NonStop is receiving the file, the remote partner determines the packing block size.

Note: The mainframe versions of CA XCOM Data Transport support a version of packing where records are separated by line end characters. On z/OS, this is specified by PACK=CRLF. The HP NonStop version of CA XCOM Data Transport does not support this record packing method. When you define an HP NonStop partner in the CA XCOM Data Transport for z/OS Destination Member (XCOMCNTL), you should code PACK=LENGTH and MAXPACK=31744.

Default: NO

XBUFFSIZE

Specifies the buffer size for a single record. Set this to the maximum record size for the transfer.

For HP NonStop records, the maximum record size is 4096.

Range: 0000 to 4096

Default: 4096

Special Feature Parameters

The special feature parameters define specific CA XCOM Data Transport for HP NonStop functions for your system. They include the following:

- Checkpoint/restart parameters
- Testing and tracing parameters
- Store-and-forward parameters
- Notification parameters
- Security parameters

Use these groupings to identify which parameters are necessary to specific CA XCOM Data Transport for HP NonStop functions, and then edit only the parameters in that group

Example:

When editing XCOMCNF to send a file from a local system to a remote system, you must configure the following two groups of parameters:

- The configuration parameters for remote destinations
- The send file parameters

Checkpoint/Restart Parameters

Checkpoint/Restart allows transfers that fail due to a line error to be restarted from the last confirmed checkpoint. This feature can be useful when transferring data over SDLC dial-out lines where the quality of the switched connections cannot be guaranteed.

The checkpoint/restart feature adds overhead that degrades performance. When using a reliable connection, such as Token-Ring, Ethernet LAN, or a channel connection using SNAXLink, the checkpoint/restart feature should be disabled by specifying CHECKPOINT_COUNT=0.

Failed transfers can be restarted manually or automatically. To manually restart a failed transfer, use the command line interface specifying the parameters REQUEST_NO=*request_number* and RESTART_FLAG=YES. To automatically restart a failed transfer, use the XCOMDMN process. For more information, see the chapter "Operation and Control."

Important! To restart failed transfers successfully (either manually or automatically), set the following parameter. This setting is also required to support checkpoint/restart:

- RETRIES=value greater than zero

CHECKPOINT_COUNT

Specifies the number of records between checkpoints.

Note: This parameter is not recognized unless VERSION=2.

Range: 0000 to 9999

Default: 0000

CHECKPOINT_FILE

Specifies the name of the checkpoint file to which the checkpoint requests are written.

Use the following format:

vol.subvol.filename

Because of changes in the layout of the transfer record to accommodate TCP/IP, when upgrading from a previous version the checkpoint file must be redefined, as follows:

```
FUP PURGE CKPTFIL
FUP PURGE CKPTALT
FUP /IN MKCKPT/
```

Range: Up to 27 characters

Default: ckptfil

RESTART_SUPPORTED

Determines whether a locally initiated transfer can be started, as follows:

- If CHECKPOINT_COUNT is zero, the transfer is restarted from the beginning.
- If CHECKPOINT_COUNT is greater than zero, and a checkpoint has been reached, the transfer is restarted from the last confirmed checkpoint.
- If RESTART_SUPPORTED=NO, the transfer is not retried, regardless of the CHECKPOINT_COUNT value.

Default: YES

Testing and Tracing Parameters

Use the following parameters for testing and tracing procedures.

For more information, see the appendix "Problem Determination."

XTRACE

Indicates the trace level.

Tracing adds considerable overhead and should be used only for problem determination. For routine transfers, specify XTRACE=1 or XTRACE=0.

Range: 0 to 9

0

No tracing.

9

Output the raw contents of data buffers.

Important! Because non-ASCII characters may be interpreted as control characters on your terminal, use the highest trace level with caution.

Default: 0

RTRACEFILE

Specifies the name and location of the trace file for remotely initiated transfers. You can specify as little of the path name as you like.

Example:

Suppose CA XCOM Data Transport for HP NonStop creates the remote trace file name as follows:

```
$SYSTEM.CAXCOM.RT971101
```

Then you could specify the following RTRACEFILEs, which would create the file names shown:

$DSV

$DSV.CAXCOM.RT971101

$DSV.XCOM

$DSV.XCOM.RT971101

$DSV.XCOM.RTRACE

$DSV.XCOM.RTRACE

Notes:

- If you specify only volume or volume/subvolume, a new file name is created when the day changes.
- If RTRACEFILE is not specified, the value specified for XDIR is used. If neither RTRACEFILE nor XDIR is specified, the default volume used is $SYSTEM. The default subvolume is as follows:

  For TCP/IP transfers

  CAXCOM

  For SNA transfers

  The LU name
- If RTRACEFILE is set to NONE, no file is created and no trace information is written.

Store-and-Forward Parameters

Use the following parameters to store-and-forward (indirectly transfer) files through a z/OS or z/VSE mainframe system.

REMOTE_SYSTEM

Specifies the name of the remote system, as defined in the CA XCOM Data Transport Destination Table on the mainframe, as follows:

- For z/OS and z/VSE, this would be the XCOMCNTL member name.

Range: Up to 128 characters

Default: XCOMAPPL

XIDEST

Indicates the intermediate destination name for indirect transfers, as follows:

- If XIDEST is null or unset, a direct connection to the remote system is attempted.
- If XIDEST contains a value, it is taken to be the name of an intermediate CA XCOM Data Transport destination that will handle traffic to and from the named remote system.

Range: Up to 21 characters

Default: None

Notification Parameters

Use the following parameters for CA XCOM Data Transport notification procedures.

LCLNTFYL

Specifies the local user notification level.

ALL

NOTIFY on transfer completion.

WARN

NOTIFY only if the transfer received a warning or error.

ERROR

NOTIFY only if the transfer received an error.

Default: ALL

LOCAL_NOTIFY

Specifies which user to notify on the local system when CA XCOM Data Transport has completed the transfer.

Range: Up to 64 characters

Default: None

NOTIFY_NAME

Specifies which user to notify on the remote system when CA XCOM Data Transport has completed its procedure.

If the remote system is a z/OS system, CA XCOM Data Transport uses the value of NOTIFYR to determine the type of notification to deliver.

If the remote system is an HP NonStop system, the user receives a mail message.

Range: Up to 12 characters

Default: None

NOTIFYR

Specifies the notification flag on the remote system.

TSO

TSO user notification.

WTO

Write to log only.

CICS

CICS user notification.

LU

Logical unit notification.

VM

VM/CMS user notification.

NONE

No user notification.

Note: This parameter is associated with the NOTIFY_NAME parameter.

Default: NONE

RMTNTFYL

Specifies the remote user notification level when sending data to a remote system.

ALL

NOTIFY on transfer completion.

WARN

NOTIFY only if the transfer received a warning or error.

ERROR

NOTIFY only if the transfer received an error.

Default: ALL

Security Parameters

For security reasons, you *must* specify the following parameters. For more information about security features, see the chapter "Security."

Note: HP NonStop user IDs and passwords are case-sensitive.

DOMAIN

The Windows domain name for use in authenticating the user ID and password when accessing a Windows based machine that has sharable disks and drives that belong to that domain. This allows users to access these sharable drives without having to have a local user ID or password defined to the machine.

Range: 1 to 15 characters

Default: None

PASSWORD

Indicates the remote password to use with the file security scheme on the remote system.

Range: Up to 31 characters

Default: None

PASSWORD_FILE

Specifies the name of the CA XCOM Data Transport security file. Use the following format:

vol.subvol.filename

Range: Up to 27 characters or NONE

NONE

Disables the CA XCOM Data Transport security feature.

Note: Setting PASSWORD_FILE=NONE disables CA XCOM Data Transport security only. It does not affect Tandem security. For more information, see the chapter "Security."

Default: NONE

USERID

Identifies the remote user ID for use with the file security scheme on the remote system.

Range: Up to 12 characters

Default: None

Scheduling Transfers Using the XCOMDMN

The following parameters can be used when scheduling transfers with the XCOMDMN.

Important! Both parameters, START_DATE and START_TIME, are required to schedule a transfer. If START_DATE and START_TIME are not specified, the transfer starts immediately.

START_DATE

Indicates the date on which the daemon should start the scheduled transfer.

The format of START_DATE depends on the setting of the EURO_DATE parameter, as follows:

EURO_DATE value = YES

The format is DD/MM/YY.

EURO_DATE value = NO

The format is MM/DD/YY.

Default: None

Note: If START_DATE and START_TIME are not specified, the transfer starts immediately.

START_TIME

Indicates the time at which the daemon should start the scheduled transfer. The format of START_TIME is HH:MM:SS, in 24-hour military time.

Default: None

Note: If START_DATE and START_TIME are not specified, the transfer starts immediately.

Gateway Parameters

This section describes CA XCOM Gateway parameters used by CA XCOM Data Transport for HP NonStop.

GATEWAYGUID

Identifies the remote file as a CA XCOM Gateway file and specifies the CA XCOM Gateway GUID. The CA XCOM Gateway GUID is a unique value that identifies each CA XCOM Gateway file. The keyword ANY can be used to identify the remote file as a CA XCOM Gateway file when the CA XCOM Gateway GUID is not known.

Range: 0 to 36 characters

Default: None (the remote file is not a CA XCOM Gateway file)

Chapter 5: The Batch and Command Line Interface

This chapter explains the command line interface parameters and syntax. It also identifies options and commands used to perform file transfers.

This section contains the following topics:

Initiate Command Line Transfers

To initiate command line transfers, you need to run the XCOM62 program in one of two ways:

- Interactively from a terminal, or
- By invoking it from a batch OBEY file

Each transfer that you initiate runs as a separate process. When the transfer completes, successfully or not, the process terminates. CA XCOM Data Transport for HP NonStop places no restrictions on the number of simultaneous transfers.

Designate Parameter Values

You can designate parameter values using:

- The PARAM function
- The SET option
- HP NonStop OBEY files
- Standard IN files

Note: Check that the TCP/IP and/or SNAX/APC processes are started before attempting to use CA XCOM Data Transport for HP NonStop.

Encrypt Configuration Files

You can encrypt specified parameters, including USERID and PASSWORD, in existing configuration files. An encrypted configuration file is used just like a non-encrypted configuration file.

Command Syntax

The basic form of a command line request is as follows:

```
[RUN] XCOM62 command filename1 AS filename2[,parameters]
```

command

Specify one of these commands:

PUT

Transfer a HP NonStop disk file to a disk file on the remote system (can use SENDFILE or SF).

GET

Transfer a disk file from a remote system to a disk file on the HP NonStop system (same as RECEIVEFILE or RF).

SR

Transfer a HP NonStop disk file to a remote system for printing (same as SENDREPORT).

SJ

Transfer a HP NonStop disk file to a remote system for execution as a job (same as SENDJOB).

filename1

For PUT, SR, and SJ commands, use the name of the local HP NonStop.

For the GET command, use the name of the file on the remote system.

filename2

For the PUT command, use the name of the file on the remote system.

For the GET command, use the name of the file on the local HP NonStop system.

parameters

Specify a string of parameters, or a parameter file name, or both, separated by commas.

There are several ways to specify parameter values:

- Default to the XCOMCNF file containing the customized default parameter values.
- Use the SET option if XCOM62 has been run with no parameters specified.
- Use the PARAM function to set any parameter before invoking CA XCOM Data Transport for HP NonStop.
- Specify the parameter value on the command line.
- Specify a configuration file on the command line.

The following is an example of a valid file transfer:

```
xcom62 get hlq.abc(xyz)as $vol.subvol.filename,remote_system=141.202.12.34
```

Specify Parameter Values on the Command Line

The following sections explain how you can override the XCOMCNF file, which contains customized default parameter values.

Override XCOMCNF

When you enter a command and file names, CA XCOM Data Transport for HP NonStop uses the parameter values from the file specified in XCOMCNF to perform the transfer. To override the XCOMCNF defaults, you can specify parameter values directly from the command line.

As shown below, enter a command followed by a comma and a parameter name with its new value:

```
[RUN] XCOM62 command local_filename as remote_filename,
parameter_name1=parameter_value1,parameter_name2=parameter_value2
```

Note: This command is shown on two lines due to the margins of the page. Make sure you type the entire command before pressing Enter, or you will execute an incomplete command.

Format for Parameter Overrides

You must use the following format for these parameter overrides:

parameter_name=parameter_value

Sample File Transfer

In the following sample file transfer, the FILE_OPTION, XMODE, REMOTE_SYSTEM, and XLUNAME values are set on the command line and override those in the XCOMCNF file.

```
xcom62 get tandem.test as
scixcom.tndmtst,file_option=REPLACE,
xmode=XCOM4K,remote_system=LUSTRATT,xluname=LUTANDEM
XCOMT0010I Starting CA-XCOM Transfer on 1991/06/17, 11:07:54
XCOMT0014I Receiving local file .tndmtst' from LUSTRATT tandem.test
XCOMT0002I Received 571 records, 16457 bytes, in 43 seconds (382 bytes/sec)
```

Run the Software Interactively

The following sections explain how you can use the SET option to enter parameters interactively.

Interactive Parameter Override Format

If you invoke CA XCOM Data Transport without specifying any parameters, CA XCOM Data Transport for HP NonStop lets you enter parameters interactively from the command line. This process is similar to how you would use the SET option in FUP. To override the XCOMCNF parameter values, use the following format:

SET *parameter_name=parameter_value*

You can set as many values as you wish, but you must end the sequence with a CA XCOM Data Transport for HP NonStop command.

Note: Before you can execute CA XCOM Data Transport for HP NonStop transfers, your TCP/IP or SNAX/APC process must be active.

Sample Interactive Commands

The following are examples of interactive command usage:

```
$xata3 chad 91>sci.xcom62
- set file_option=REPLACE
- set userid=GREEN
- set password=MIKE
- put scixcom.tndmtst as tandem.test
XCOMT0010I Starting CA-XCOM Transfer on 1997/06/17, 11:19:45
XCOMT0015I Sending local file .tndmtst' to LUSTRATT tandem.test
XCOMT0001I Sent 571 records, 15806 bytes, in 42 seconds (376 bytes/second)
- EXIT
$xata3 chad 92>
```

HP NonStop SET Option Commands

Use the following table to identify the CA XCOM Data Transport for HP NonStop SET option commands:

Command	Syntax	Purpose
EXIT	-EXIT	Returns control to Guardian.
FC or fixed command	-FC	HP NonStop convention for editing the last line of input.
GET or receivefile or RF	-RF *remote_filename* as *local_filename*	Retrieves a file.
OBEY or O	-OBEY *obey_filename* or o obey_filename	Invokes HP NonStop OBEY files.
PUT or sendfile or SF	-PUT *local_filename* as *remote_filename*	Sends a file.

The PARAM Function

The following sections explain how you can use the PARAM function to set parameter values, or override XCOMCNF.

PARAM Function Syntax

To set parameter values using Tandem's PARAM function

Use the following syntax:

```
PARAM parameter_name parameter_value
```

To designate a configuration file to override XCOMCNF

Use the following syntax:

```
PARAM XCOMCNF configuration_filename
```

Note: Tandem does not allow the use of an underscore in the PARAM field. When the parameter_name contains an underscore, substitute a hyphen for the underscore. For example, to set the parameter LOCAL_FILE, use the following syntax:

```
PARAM LOCAL-FILE $<vol>.<subvol>.<file>
```

Sample PARAM Override

In the following example, the PASSWORD value set with the PARAM command overrides the PASSWORD setting in the XCOMCNF configuration file. However, parameter values set by the PARAM function do not override those parameter values set on the command line or set in other configuration files invoked on the command line.

```
param  password xxxxx
$xata3 chad 90>sci.xcom62 get tandem.test as scixcom.tndmtst
XCOMT0010I Starting CA-XCOM Transfer on 1991/06/17, 11:14:20
XCOMT0014I Receiving local file .tndmtst' from LUSTRATT tandem.test
XCOMT0002I Received 571 records, 16457 bytes, in 53 seconds (310 bytes/sec)
```

Note: The values set with PARAM stay in effect until they are cleared. For example, if you are using SNAX, the SNAX startup uses the PARAM function for setting the SNAX password. Unless this is cleared, this PARAM value overrides the CA XCOM Data Transport for HP NonStop configuration file's PASSWORD setting.

Using OBEY Files and the OBEY Command

OBEY Command Syntax

You can create OBEY files that contain commands and parameter values specific to an individual transfer. CA XCOM Data Transport for HP NonStop provides an OBEY command for invoking HP NonStop OBEY files that uses the following syntax:

OBEY *obey_filename*

or

o *obey_filename*

Example:

If at the end of every week you want to replace a file on an IBM mainframe, you can place the parameter values and the send file command necessary for this transfer in an OBEY file.

Then, at the end of the week, you would invoke the OBEY file on the command line using the OBEY command according to the syntax above.

The command in the OBEY file performs the transfer, and for that particular transfer the parameter values in the OBEY file override those in XCOMCNF (which might be set to perform routine daily transfers).

Sample OBEY Command and File

The following sample shows how CA XCOM Data Transport for HP NonStop invokes an OBEY file named IBMSND on the command line:

```
$xata3 chad 91 sci.xcom62
- obey IBMSND
XCOMT0010I Starting CA-XCOM Transfer on 1991/06/17, 12:01:05
XCOMT0015I Sending local file 'scixcom.t55' to TS223 da1mg62.t55
XCOMT0001I Sent 571 records, 15806 bytes, in 42 seconds (376 bytes/second)
                .
                .
                .
- EOF!
```

The following is the sample OBEY file IBMSND itself:

```
put t55 as da1mg62.t55
put t56 as da1mg62.t56
get da1mg62.t55 as t653
```

About Configuration Files

The following sections explain how to use configuration files.

Specify Configuration Files on the Command Line

If CA XCOM Data Transport for HP NonStop finds a comma without the equal sign denoting a parameter value on the command line, the program considers the entry following the comma to be a configuration file name. For example, the following command instructs the program to send a file using the parameter settings in the OVERDS file:

```
[RUN] XCOM62 PUT local-file-name as remote-file-name,OVERDS
```

You can use this handy feature of CA XCOM Data Transport for HP NonStop to create configuration files for specific remote systems and/or transfers.

Configuration Files for Remote Systems

If you have two different APPC connections to your mainframe and you want to send reports using either connection, you might have two files.

Example File:

A file named MVSSNAX that contains parameters for one type of APPC connection to the remote system:

```
REMOTE_SYSTEM=TS222
XMODE=XCOMMODE
XLUNAME=LU338B02
APPC_PROCESS_NAME=$SNAS
APPC_OPEN_NAME=
APPC_TYPE=SNAXAPPC
# Parameters for sending reports
DESTINATION=
USERID=DA1MG62
PASSWORD=GREEN
```

Configuration Files for Specific Transfers

To send the same type of report transfer to a number of different remote systems, make sure your configuration file contains only parameter values for the transfer itself (for example, class, copies, report_title). This ensures that the various remote systems are consistently getting the same information, which you only have to type out once.

Example:

Here is an example of such a configuration file for a report transfer:

```
FILE_OPTION=REPLACE
CLASS=A
COPIES=025
```

Configuration and Parameter Priority

It is possible to make a transfer request that invokes conflicting parameter and/or configuration options. In all cases, the parameter entered last, whether in-line or in a configuration or OBEY file, takes precedence.

For example, let's say the configuration file MVSPARMS has the FILE_OPTION parameter set to REPLACE. In the transfer below, the parameter value specified in MVSPARMS would override the FILE_OPTION parameter value specified on the command line as APPEND.

```
RUN XCOM62 PUT tandem.test as xcom.filetest, FILE_OPTION=APEND, MVSPARMS
```

Note: This command is shown on two lines due to the margins of the page. Make sure you type the entire command before pressing Enter, or you will execute an incomplete command.

If, however, the order of the configuration file and the specified parameter value were reversed, CA XCOM Data Transport for HP NonStop would use the specified command line value APPEND for the value of FILE_OPTION.

Commands and Command Line Syntax

CA XCOM Data Transport for HP NonStop uses four commands to execute file transfers. These commands are listed in the table below:

Command	Syntax	Purpose
PUT or sendfile or SF	[RUN] XCOM62 PUT *local_filename* as *remote_filename*	Used to send a file.
GET or receivefile or RF	[RUN] XCOM62 RF *remote_filename* as *local_filename*	Used to retrieve a file.
SR or sendreport	[RUN] XCOM62 SR *local_filename*	Used to send a report.
SJ or sendjob	[RUN] XCOM62 SJ *local_filename*	Used to send a job.

Note: Because CA XCOM Data Transport for HP NonStop commands are not case-sensitive, the commands listed above can be entered in lowercase or uppercase characters.

Send File Transfers

Use the following syntax to send a file:

```
[RUN] XCOM62 command local_filename as remote_filename, parameter1=value
```

The Send File syntax parameters are as follows:

command

Use either PUT, SENDFILE, or SF.

local_filename

The name of the file on the local system that you wish to send.

remote_filename

The name of the file on the remote system to which you want to create, replace, or append.

parameter1=value

This value overrides the parameter setting in the XCOMCNF file.

Sample Send File Transfer

In the Send File Transfer command example shown below, TANDEM.TEST is the local file name, XCOM.TESTSEND is the remote file name, and FILE_OPTION=REPLACE is the command line parameter setting override.

```
[RUN] XCOM62 SF TANDEM.TEST as XCOM.TESTSEND, FILE_OPTION=REPLACE
```

Retrieve File Transfers

To retrieve a file, use the following syntax:

```
[RUN] XCOM62 command remote_filename as local_filename, parameter1=value
```

The Retrieve File parameters are as follows:

command

Use either GET, RF, or receivefile.

remote_filename

The name of the file on the remote system that is to be transferred.

local_filename

The name of the file on the local system you wish to create, replace, or append to.

parameter1=value

This setting overrides the parameter setting in the XCOMCNF file.

Sample Retrieve File Transfer

In the Retrieve File transfer shown below, TANDEM.TESTRECV is the remote file name, XCOM.TANTEST is the local file name, and FILE_OPTION=REPLACE is the command line parameter setting override.

```
[RUN] XCOM62 RF TANDEM.TESTRECV as XCOM.TANTEST, FILE_OPTION=REPLACE
```

This command is shown on two lines due to the margins of the page. Make sure you type the entire command before pressing Enter, or you will execute an incomplete command.

Send Report Transfers

To send a report, use the following command line syntax:

```
[RUN] XCOM62 command local_filename, parameter1=value
```

The Send Report parameters are as follows:

command

Use either SR or sendreport.

local_filename

The name of the remote on the local system that you wish to send.

parameter1=value

This setting overrides the parameter setting in the XCOMCNF file.

Sample Send Report Transfer

In the Send Report command example shown below, TANDEM.REPORT is the local file name and COPIES, HOLDFLAG, and REPORT_TITLE are parameter values set on the command line.

```
[RUN] XCOM62 SR TANDEM.REPORT,COPIES=3,HOLDFLAG=YES, REPORT_TITLE=TANDEM
```

Send Job Transfers

To send a job to a remote system, use the following syntax:

```
[RUN] XCOM62 command local_filename, parameter1=value
```

The Send Job parameters are as follows:

command

Use either SJ or sendjob.

local_filename

The name of the local system job that you want to execute on the remote system. The job must contain the control statements necessary to execute the job on the remote system.

parameter1=value

This setting overrides the parameter setting in the XCOMCNF file.

Sample Send Job Transfer

In the Send Job command example shown below, TANDEM.JOB is the local file name and COPIES and HOLDFLAG are parameters defined on the command line.

```
[RUN] XCOM62 SJ TANDEM.JOB,COPIES=3,HOLDFLAG=YES
```

Batch Processing

The following sections explain how to use batch processing.

Standard I-completion Structure

CA XCOM Data Transport for HP NonStop supports TACL batch processing by returning completion codes for TACL to place in the standard i-completion structure shown in the following table:

```
[#DEF :_ completion STRUCT
  BEGIN
  INT       messagecode;
  CRTPID    process;
  INT       headersize VALUE 14;
  INT4      cputime;
  INT       jobid;
  INT       completioncode;
  STRUCT    internal;
    BEGIN
    INT     terminationinfo;
    SSID    subsystem;
    END;
  STRUCT    external REDEFINES internal;
    BEGIN
    BYTE    group;
    BYTE    user;
    CRTPID  process;
    END;
  INT       textlength;
  CHAR      tex(0:79);
  END;
```

After a CA XCOM Data Transport for HP NonStop transfer completes, CA XCOM Data Transport for HP NonStop fills in the value of COMPLETIONCODE.

If the transfer was successful, COMPLETIONCODE has the value of 0. If the transfer failed, the ABEND code is returned by CA XCOM Data Transport for HP NonStop.

Note: Although TACL macros can use COMPLETIONCODE to determine if CA XCOM Data Transport for HP NonStop transfers were successful, COMPLETIONCODE cannot be used to determine what type of error caused a transfer to fail.

Sample Completed I-Completion Structure

In the following example, the i-completion structure has been filled in by CA XCOM Data Transport for HP NonStop:

```
$CLX12 SCI 66 outvar _completion
$CLX12 SCI 66..
_COMPLETION(0)
  MESSAGECODE (0:0)
                 -6
  PROCESS(0:0)    5,42
  HEADERSIZE(0:0) 14
  CPUTIME(0:0)    5901158
  JOBID(0:0)      0
  COMPLETIONCODE(0:0)
                 5
  INTERNAL(0)
    TERMINATIONINFO(0:0)
                   0
    SUBSYSTEM(0:0)  0.0.0
  TEXTLENGTH(0:0) 0
  TEXT(0:79)
$CLX12 SCI 67
```

Sample TACL Macro

In the following example, the sample TACL macro performs a CA XCOM Data Transport for HP NonStop transfer. If the first transfer succeeds, the next transfer is run.

```
?tacl macro
#FRAME
run xcom62 put t55 as da1mg62.mvst55,qacnf,password=SKIP,userid=DA1MG62
[#IF :_completion:COMPLETIONCODE = 0 |then|
run xcom62 put t55 as da1mg62.mvst55,qacnf ]
[#IF :_completion:COMPLETIONCODE = 0 |then|
run xcom62 put t55 as da1mg62.mvst55,qacnf ]
#UNFRAME
```

Sample TACL Macro for TCP/IP

The following example shows a sample TACL macro for performing a CA XCOM Data Transport for HP NonStop transfer using TCP/IP:

```
?tacl macro
#FRAME
SINK #DEFINEDELETE = _EMS_COLLECTOR
SINK #DEFINE = _EGEN_ADD_EVENT_TEXT
SINK #DEFINEDELETE = _EMS_COLLECTOR, CLASS MAP, FILE $0
SINK #DEFINE = _EGEN_ADD_EVENT_TEXT, CLASS MAP, FILE $YES

PARAM XCOMCNF $DAT0.XCOMTEST.XCOMCNF

ADD DEFINE=TCPIP^HOST^FILE, CLASS MAP, FILE $DAT0.XCOMTEST.HOSTS

RUN XCPTIOBJ.XCOM62/name/SF $DAT0.XCOMTEST.EDITFILE AS & $DAT1.XCOMTEST.EDITFILE

CLEARALL
TIME
#UNFRAME
```

Encrypt Parameter Values in Existing Configuration Files

Use XCOMENCR to encrypt selected parameter values up to 31 characters in an existing configuration file. XCOMENCR is intended for encrypting the USERID and PASSWORD parameters, but all parameters can be encrypted. If a parameter value is greater than 31 characters, only the first 31 characters are encrypted and a comment line indicating this is placed in the configuration file above the encrypted parameter. Null parameter values are not encrypted.

About Encrypting Parameter Values

When XCOMENCR encounters a line with #!ENCRYPT, it changes the next non-comment line from:

`PARAMETER=VALUE`

To:

`PARAMETER.ENCRYPTED=ENCRYPTEDVALUE`

Example:

Before encryption:

`PASSWORD=ENIGMA`

After encryption:

`PASSWORD.ENCRYPTED=12 0F 36 79 65 AB D0 37 ...`

up to 32 hexadecimal characters.

Syntax

The syntax for using XCOMENCR is as follows:

`[RUN] XCOMENCR input_file <option>`

Options

The following list explains the options for XCOMENCR:

Output_file

Send output to another file

- (minus sign)

Send output to stdout.

Example: XCOMENCR *input_file* **-**

+ (plus sign)

Replace input_file.

Example: XCOMENCR *input_file* **+**

no options

Displays help text.

Procedure

To encrypt a parameter value using XCOMENCR

1. Using a text editor, open the configuration file you want to modify, go to the parameter you want to encrypt, create a blank line above it, and type in the following:

   ```
   #!ENCRYPT
   ```

   Repeat for each parameter you want to encrypt.

   Note: Because # denotes a comment line in the xcomcnf file, any line beginning with #!ENCRYPT is ignored by CA XCOM and is be used by XCOMENCR only.

2. Save the configuration file as an edit file.

3. At the command prompt, type the following and press ENTER:

   ```
   [RUN] XCOMENCR input_file output_file
   ```

   Replace *input_file* with the name of the configuration file from step 1.

   The parameter value in the first non-comment line after each occurrence of the #!ENCRYPT statement is changed to the encrypted parameter value format.

   Note: If you open an encrypted configuration file with a text editor, you would not be able to see the values of the encrypted parameters.

Change an Encrypted Value

To change an encrypted value that is already specified

Delete all text after the parameter name and give it a new value.

Example:

Suppose the line you want to change is:

```
PASSWORD.ENCRYPTED=12 0F 36 79 65 AB D0 37 ...
```

Then you have to delete:

```
.ENCRYPTED=12 0F 36 79 65 AB D0 37 ...
```

Then type in an equal sign and the new parameter value, (in an unencrypted form) replacing *NEWVALUE* with your desired value, as follows:

```
PASSWORD=NEWVALUE
```

Then save the file as an edit file and encrypt it using the encryption procedure.

Chapter 6: The Application Programming Interface

CA XCOM Data Transport for HP NonStop provides an Application Programming Interface (API) that allows applications to initiate transfers. The CA XCOM Data Transport for HP NonStop API allows for Tandem I/O program features, including Wait and Nowait I/O. You can use any programming language, including TAL, COBOL, Fortran, Pascal, and C to call CA XCOM Data Transport for HP NonStop.

This section contains the following topics:

API Version

An API is provided to allow integration of CA XCOM Data Transport for HP NonStop into client applications. The API has changed from previous versions. If you wish to use your old application with CA XCOM Data Transport for HP NonStop r11, you must recompile with the new API structures and bind your program with the new APIO file. The apiversion field must be set to 3 or CA XCOM Data Transport for HP NonStop will reject your transfer with an error 520. If the version field is not set by your program, CA XCOM Data Transport for HP NonStop uses the configuration file set by your program. If none is given, it will default to 2. If CA XCOM Data Transport for HP NonStop has a problem loading your program's API parameters, it will return an appropriate error to your program.

Note: Check that the SNAX/APC and/or TCP/IP processes are started before attempting to use CA XCOM Data Transport for HP NonStop.

The API Call

Applications invoke CA XCOM Data Transport for HP NonStop transfers by calling the following procedure:

```
XCOM62^API
```

Because XCOM62^API is written in TAL, you can invoke it from programs written in any language.

XCOM62^API is delivered in object form as the file APIO. Therefore, your application must be bound using APIO. This can be done by compiling your program as executable and including a search for APIO or using the BINDER.

XCOM62^API creates a process that runs XCOM62. In order for XCOM62^API to locate the XCOM62 object, you must have a DEFINE statement for XCOM62-PROGRAM. Define XCOM62-PROGRAM with the file parameter set to the XCOM62 object as follows:

```
ADD DEFINE =XCOM62-PROGRAM, CLASS MAP, FILE vol.subvol.XCOM62
```

Startup Messages

When HP NonStop processes create new processes in a HP NonStop environment, special startup messages must be sent to the new processes. XCOM62^API takes care of this for your application and sends a message to the XCOM62 process containing the api-transfer structure.

The XCOM API supports Nowait I/O. If Nowait I/O is turned on (the NOWAIT^IO parameter is set to TRUE), XCOM62^API can return immediately to your application without waiting for a reply message from CA XCOM Data Transport for HP NonStop. If Nowait I/O is not turned on (NOWAIT^IO = FALSE), the application waits for the transfer completion status reply from CA XCOM Data Transport for HP NonStop.

For more information about using Nowait I/O, see the *Guardian 90 Operating System Programming Guide.*

Error Messages

An error message returned from the API call can be either of two things:

- A CA XCOM Data Transport for HP NonStop error message as defined in the API structure.
- An error message returned from the Tandem NEWPROCESS call.

For example, a common error number returned from the API call is 966. This is a NEWPROCESS error message that indicates that you did not define where the XCOM62 program resides.

For a detailed description of NEWPROCESS errors, see the *Tandem PROC Calls* manual.

External Declaration Statement

To call CA XCOM Data Transport for HP NonStop's API, use the following external declaration statement:

```
SECTION xcom62^api
INT  PROC XCOM62^API (transfer, pid, nowait^io, file^num, startup^only) VARIABLE;
  INT .EXT transfer (api^transfer^def);
  INT .EXT pid;
  INT nowait^io;
  INT .EXT file^num;
  INT startup^only;    -- If this is true, do not write API structure
                       -- to the NEWPROCESSed XCOM. Leave that
                       -- to the calling application.
```

Parameter Descriptions

The following parameters are included in the external declaration statement.

transfer

Specifies the file structure that contains the file transfer information for your program. Use APIDDL to generate language-specific file structures.

The following files are included on the distribution media:

File	Language
XAPIC	C language
XAPICOB	COBOL
XAPIFOR	Fortran

XAPIPAS	Pascal
XAPITAL	TAL
XAPITACL	TACL

For these transfer structures, you specify the parameter values for each CA XCOM Data Transport for HP NonStop parameter that you require for the transfer. The fields in the structure are described in the following pages and correspond to the parameters defined in the chapter "Configuring CA XCOM Data Transport for HP NonStop."

pid

XCOM62^API returns the process ID of the XCOM62 process that it created to do the transfer.

nowait^io

If you specify a non-zero value here, XCOM62^API will return to the user application before receiving a status reply from the XCOM62 process. If you specify zero, XCOM62^API will wait for the status reply. The application is responsible for receiving the CA XCOM Data Transport for HP NonStop reply.

file^num

Indicate the file number that XCOM62^API returns for the XCOM62 process.

If the NOWAIT^IO value is non-zero, this file number receives messages from CA XCOM Data Transport for HP NonStop. For more information about using Nowait I/O, see the Guardian 90 Operating System Programming Guide.

startup^only

Set by the calling process. If this is true, do not write API structure to the NEWPROCESSed XCOM.

Data Dictionary Language (DDL) Input Statements

The following sections describe the DDL input statements.

File Format

Use the Tandem DDL commands to generate language-specific files from the APIDDL file. The following file shows the format of the message sent from an application program to the CA XCOM Data Transport Application Programming Interface:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!    Data Dictionary Language input statements for CA-XCOM Tandem API
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

?dictn XCTNDM11!
?SAVE
?COMMENTS
?DEFLIST
?VALUES
?WARN
?ERRORS 30
?NOTIMESTAMP
!
!
?TALBOUND 0
?TALCHECK
?TAL XCTNDM11.XAPITAL  !
!
?CLISTIN
?CLISTOUT
?CLISTOUTDETAIL
?NOCDEFINEUPPER
?CCHECK
?C XCTNDM11.XAPIC    !
!
?COBCHECK
?SETCOBOL85
?COBOL XCTNDM11.XAPICOB !
!
?FORTRANUNDERSCORE
?FORCHECK
?FORTRAN  XCTNDM11.XAPIFOR  !
!
?PASCALCHECK
?PASCAL XCTNDM11.XAPIPAS  !
!
?TACL XCTNDM11.XAPITACL  !
?NOFUP


?SETSECTION api-transfer
```

```
*************************************************************************
*
*    This file was generated from apiddl
*
*************************************************************************

DEF    api-transfer .

! GENERAL PARAMETERS
 02 apiversion       TYPE BINARY     16.  ! api version number                !
 02 configfile       TYPE CHARACTER  36.  ! name of config file to use        !
 02 wait             TYPE BINARY     16.  ! wait for transfer to complete?    !

! DESTINATION CONFIGURATION PARAMETERS
 02 appcopenname     TYPE CHARACTER  16.  ! additional qualifiers for ice open !
 02 appcprocessname  TYPE CHARACTER  16.  ! file name used to open ice        !
 02 appctype         TYPE CHARACTER   1.  ! S - SNAX, T - TCP/IP              !
 02 command          TYPE BINARY     16.  ! XCOM command code                 !
 02 idest            TYPE CHARACTER 128.  ! intermediate destination          !
 02 luname           TYPE CHARACTER  18.  ! name of local LU                  !
 02 max-snax-iosize  TYPE CHARACTER   6.  ! Snax apc buffer size              !
 02 rluname          TYPE CHARACTER 128.  ! name of remote LU or TCP/IP name  !
 02 version          TYPE BINARY     16.  ! XCOM version #                    !
 02 xdir             TYPE CHARACTER  25.  ! Default XCOM directory            !
 02 xlogfile         TYPE CHARACTER  36.  ! name of log file to use           !
 02 xmode            TYPE CHARACTER   9.  ! mode used by XCOM                 !

! SEND AND RETRIEVE FILE PARAMETERS
 02 ascebc           TYPE CHARACTER  27.  ! file to use for ASCII to EBCDIC   !
 02 bulkio           TYPE CHARACTER   2.  ! bulkio buffersize 2,4,...,30      !
                                          ! if zero bulkio not in use         !
 02 cache-buf        TYPE CHARACTER   1.  ! set cache buffering Y/N           !
 02 checkpoint-count TYPE CHARACTER   5.  ! checkpoint count                  !
 02 checkpoint-file  TYPE CHARACTER  27.  ! checkpoint file name              !
 02 compression      TYPE CHARACTER   1.  ! compression                       !
 02 ebcasc           TYPE CHARACTER  27.  ! file to use for EBCDIC to ASCII   !
 02 fileaction       TYPE CHARACTER   1.  ! file action                       !
 02 history-file     TYPE CHARACTER  27.  ! history  file name                !
 02 ipc-pname        TYPE CHARACTER  25.  ! IPC process name to read/write to !
 02 ipc-fname        TYPE CHARACTER 151.  ! IPC filename to run if IPC process !
                                          ! does not exist                    !
 02 nullfill         TYPE CHARACTER   1.  ! null fill flag for text files     !
 02 pack             TYPE CHARACTER   1.  ! set packing Y/N                   !
 02 request-no       TYPE CHARACTER   7.  ! unique # range 1..999,999         !
 02 rfile            TYPE CHARACTER 255.  ! name of remote file               !
 02 restart-flag     TYPE CHARACTER   1.  ! Y/N, yes it's a restart           !
 02 sio              TYPE CHARACTER   1.  ! set sio calls for edit files Y/N  !
 02 transfer-id      TYPE CHARACTER  11.  ! non-unique transfer id            !
 02 xbuffsize        TYPE CHARACTER   7.  ! buffer size for a single record   !
```

```
 02 xfile            TYPE CHARACTER 255.  ! name of local file                !

! DISK FILE CREATION PARAMETERS
 02 carriageflag     TYPE CHARACTER   1.  ! carriageflag Y/N                   !
 02 codeflag         TYPE CHARACTER   1.  ! A - ASCII, B - Binary, E - EBCDIC  !
 02 guardianfiletype TYPE CHARACTER   2.  ! Tandem filetype: EDit file         !
                                          !       ENtry sequenced file         !
                                          !       RElative file                !
                                          !       UNstructured file            !
 02 prialloc         TYPE CHARACTER   6.  ! primary allocation                !
 02 recfm            TYPE CHARACTER   4.  ! record format, VB, FB, F, U       !
 02 secalloc         TYPE CHARACTER   6.  ! secondary allocation              !

! IBM MAINFRAME FILE CREATION PARAMETERS
 02 alloc-unit       TYPE CHARACTER   1.  ! B - Block, C- Cylinder, T - Track  !
 02 blksize          TYPE CHARACTER   6.  ! physical block size               !
 02 lrecl            TYPE CHARACTER   6.  ! logical record size               !
!   recfm                                  record format                      !
 02 system-user-data TYPE CHARACTER   11. ! system dependent user data        !
 02 transfer-user-data TYPE CHARACTER 11. ! user specified data               !
 02 volume           TYPE CHARACTER   11. ! volume to create file on          !
 02 xunit            TYPE CHARACTER   11. ! unit to create file on            !

! SEND REPORTS PARAMETERS
 02 carriagecontrol  TYPE CHARACTER    1. ! A - ASA, M - IBM Machine, " "- None!
 02 chars            TYPE CHARACTER    5. ! font for report sent to MVS       !
 02 copies           TYPE CHARACTER    4. ! number of copies to be printed    !
 02 disposition      TYPE CHARACTER    1. ! D - Delete, H - Hold, K - Keep    !
 02 eol-classes      TYPE CHARACTER  128. ! print classes having NL added at  !
                                          ! end of record                     !
 02 fcb              TYPE CHARACTER    5. ! Form Control Block                !
 02 form             TYPE CHARACTER   11. ! name of special forms to used     !
 02 holdflag         TYPE CHARACTER    1. ! flag for MVS spoolers             !
 02 reporttitle      TYPE CHARACTER   22. ! report name to be printed in banner!
 02 spoolflag        TYPE CHARACTER    1. ! Y/N, send report to spooler?      !
 02 xclass           TYPE CHARACTER    1. ! print job class                   !
 02 xdestination     TYPE CHARACTER   22. ! name of remote printer            !

! JOB PARAMETERS
 02 jobname          TYPE CHARACTER    9. ! job name from remote system       !
 02 jobnumber        TYPE CHARACTER    9. ! job number from remote system     !
 02 uic              TYPE CHARACTER   10. ! not used                          !
 02 xwhen            TYPE CHARACTER   15. ! not used                          !

! STORE AND FORWARD PARAMETERS
!   idest                                  intermediate destination          !
!   rluname                                final destination                 !

! NOTIFICATION PARAMETERS
```

```
 02 localnotify      TYPE CHARACTER   65. ! name of users to notify           !
 02 notify           TYPE CHARACTER    1. ! notify flag                       !
 02 tso-notify       TYPE CHARACTER    9.
 02 who              TYPE CHARACTER   13. ! who to notify                     !

! SPOOLING PARAMETERS
 02 spoolcollector   TYPE CHARACTER   25. ! Spool collector processname       !

! SECURITY PARAMETERS
 02 conv-security    TYPE CHARACTER    1. ! set conv-security Y/N             !
 02 password         TYPE CHARACTER   32. ! password                         !
 02 password-file    TYPE CHARACTER   27. ! password file name               !
 02 remoteuser       TYPE CHARACTER   13. ! remote user id                   !

END .


?SETSECTION api-commands
*******************************************************************************
*
*    This file was generated from apiddl
*
*******************************************************************************


CONSTANT xcom-send-file      VALUE 1 .
CONSTANT xcom-send-report    VALUE 2 .
CONSTANT xcom-send-job       VALUE 3 .
CONSTANT xcom-receive-file   VALUE 4 .


?SETSECTION api-error-codes
*******************************************************************************
*
*    This file was generated from apiddl
*
*******************************************************************************

*   Error codes returned by XCOM62^API

CONSTANT ERROR-ALLOCATING-SEND-BUFFER      VALUE 283 .
CONSTANT ERROR-FORKING                     VALUE 284 .
CONSTANT ERROR-CREATING-PIPE               VALUE 285 .
CONSTANT ERROR-SETTING-LOCAL-USER-ID       VALUE 286 .
CONSTANT ERROR-SETTING-REMOTE-USER-ID      VALUE 287 .
CONSTANT ERROR-SYSTEM-FAILED               VALUE 288 .
CONSTANT ERROR-COMMAND-FAILED              VALUE 289 .
CONSTANT ERROR-RECEIVING-OVERLAY           VALUE 290 .
CONSTANT ERROR-SENDING-OVERLAY             VALUE 291 .
```

```
CONSTANT ERROR-SENDING-ERROR                VALUE 292 .
CONSTANT ERROR-EXPECTING-SEND-STATE         VALUE 293 .
CONSTANT ERROR-EXPECTING-RECEIVE-STATE      VALUE 294 .
CONSTANT ERROR-COMMAND-LINE                 VALUE 295 .
CONSTANT ERROR-DEALLOCATING                 VALUE 296 .
CONSTANT ERROR-REQUESTING-HEADR-CONFIRM     VALUE 297 .
CONSTANT ERROR-ALLOCATE                     VALUE 298 .
CONSTANT ERROR-LOCAL-ATTACH                 VALUE 299 .
CONSTANT ERROR-REMOTE-ATTACH                VALUE 300 .
CONSTANT ERROR-STARTING-APPC                VALUE 301 .
CONSTANT ERROR-OPENING-INPUT-FILE           VALUE 302 .
CONSTANT ERROR-SENDING-HEADER               VALUE 303 .
CONSTANT ERROR-SENDING-MAXLRECL             VALUE 304 .
CONSTANT ERROR-RECEIVING-HEADER             VALUE 305 .
CONSTANT ERROR-INVALID-HEADER               VALUE 306 .
CONSTANT ERROR-INVALID-PACK-OPTION          VALUE 307 .
CONSTANT ERROR-READING-INPUT-FILE           VALUE 309 .
CONSTANT ERROR-RECEIVE-ERROR                VALUE 310 .
CONSTANT ERROR-SENDING-DATA                 VALUE 311 .
CONSTANT ERROR-CONFIRMING-DATA              VALUE 312 .
CONSTANT ERROR-NEGATIVE-DATA-CONFIRM        VALUE 313 .
CONSTANT ERROR-SENDING-TRAILER              VALUE 314 .
CONSTANT ERROR-NEGATIVE-TRAILER-CONFIRM     VALUE 315 .
CONSTANT ERROR-RECEIVED-FROM-REMOTE         VALUE 316 .
CONSTANT ERROR-RECEIVE-FMH-7                VALUE 317 .
CONSTANT ERROR-REVERSING-LINE               VALUE 318 .
CONSTANT ERROR-CONFIRMING-CHECKPOINT        VALUE 319 .
CONSTANT ERROR-CONFIRMED-CHECKPOINT         VALUE 320 .
CONSTANT ERROR-OPENING-CHECKPOINT-FILE      VALUE 321 .
CONSTANT ERROR-WRITING-CHECKPOINT-FILE      VALUE 322 .
CONSTANT ERROR-RECV-RECEIVING-HEADER        VALUE 401 .
CONSTANT ERROR-HEADER-INVALID               VALUE 402 .
CONSTANT ERROR-OPENING-OUTPUT-FILE          VALUE 403 .
CONSTANT ERROR-CONFIRMING-HEADER            VALUE 404 .
CONSTANT ERROR-RECEIVING-MAXLRECL           VALUE 405 .
CONSTANT ERROR-RECEIVING-FEATURE-RECORD     VALUE 406 .
CONSTANT ERROR-MAXLRECL-INVALID             VALUE 407 .
CONSTANT ERROR-FEATURE-RECORD-PROTOCOL      VALUE 408 .
CONSTANT ERROR-REQUESTING-FEATURE-CNFRM     VALUE 409 .
CONSTANT ERROR-SENDING-FEATURE-RECORD       VALUE 410 .
CONSTANT ERROR-CONFIRM-FEATURE-RECORD       VALUE 411 .
CONSTANT ERROR-RECEIVING-DATA               VALUE 412 .
CONSTANT ERROR-TRAILER-INVALID              VALUE 413 .
CONSTANT ERROR-CLOSING-OUTPUT-FILE          VALUE 414 .
CONSTANT ERROR-RECEIVING-TRAILER            VALUE 415 .
CONSTANT ERROR-WRITING-OUTPUT-FILE          VALUE 416 .
CONSTANT ERROR-CONFIRMING-TRAILER           VALUE 417 .
CONSTANT ERROR-INTERRUPT-RECEIVED           VALUE 418 .
CONSTANT ERROR-USER-NOT-FOUND               VALUE 419 .
```

```
CONSTANT ERROR-USER-ID-OUT-OF-RANGE        VALUE 420 .
CONSTANT ERROR-GROUP-ID-OUT-OF-RANGE       VALUE 421 .
CONSTANT ERROR-LOGIN-INCORRECT             VALUE 422 .
CONSTANT ERROR-RECEIVE-FILE-DESCRIPTOR     VALUE 423 .
CONSTANT ERROR-SENDING-FILE-DESCRIPTOR     VALUE 424 .
CONSTANT ERROR-REPOSITIONING-FILE          VALUE 425 .
CONSTANT ERROR-RESTART-CNTS-DONT-MATCH     VALUE 426 .
CONSTANT ERROR-STARTING-TP                 VALUE 427 .
CONSTANT ERROR-NO-SESSIONS-AVAILABLE       VALUE 435 .
CONSTANT ERROR-TP-ABENDED                  VALUE 436 .
CONSTANT ERROR-INPUT-BUFFER-TOO-SMALL      VALUE 437 .
CONSTANT ERROR-TRUNCATION-NOT-ALLOWED      VALUE 438 .
CONSTANT ERROR-MAXLRECL-TOO-BIG            VALUE 440 .
CONSTANT ERROR-REQNOS-DONT-MATCH           VALUE 441 .
CONSTANT ERROR-FILENAMES-DONT-MATCH        VALUE 442 .
CONSTANT ERROR-GROUPNAMES-DONT-MATCH       VALUE 443 .
CONSTANT ERROR-INVALID-PACKING-TYPE        VALUE 448 .
CONSTANT ERROR-IN-COMPRESSION-TYPE         VALUE 449 .
CONSTANT ERROR-MEMORY-ALLOCATION           VALUE 468 .
CONSTANT ERROR-FILE-ALREADY-EXISTS         VALUE 470 .


CONSTANT ERROR-IPC-PARAMS-DONT-MATCH       VALUE 501 .
CONSTANT ERROR-IPC-PROGRAM-NOT-STARTED     VALUE 502 .
CONSTANT ERROR-IPC-PROCESS-NOT-EXIST       VALUE 503 .
CONSTANT ERROR-IPC-PROCESS-NAME-ILLEGAL    VALUE 504 .
CONSTANT ERROR-IPC-PROGRAM-NAME-ILLEGAL    VALUE 505 .
CONSTANT ERROR-IPC-PROCESS-ERROR           VALUE 506 .
CONSTANT ERROR-IPC-DATA-REC-LG-TOO-BIG     VALUE 507.


CONSTANT ERROR-MISSING-PARAMETER           VALUE 519 .
CONSTANT ERROR-API-INCORRECT-VERSION       VALUE 520 .
```

API General Parameters

The following sections describe the general API parameters.

APIVERSION

Indicates which API version number you are using. For this version, enter the value 3.

CONFIGFILE

Specifies the default configuration file. If this value is left blank, the default is XCOMCNF.

WAIT

Specifies whether CA XCOM Data Transport for HP NonStop should wait for the status of a transfer before sending a reply to the application program.

If you want CA XCOM Data Transport for HP NonStop to send a reply reflecting the status of the requested transfer, enter a non-zero value here.

Note: A zero causes CA XCOM Data Transport for HP NonStop to reply immediately that a transfer request has been received, but it does not inform you of the status of that transfer.

API Remote Destination Configuration Parameters

The following sections describe the remote destination configuration parameters for the API.

APPCPROCESSNAME

Specifies the name of the SNAX/APC process used by CA XCOM Data Transport for HP NonStop. This name must agree with the process name specified in the APPC configuration process.

APPCTYPE

Specifies the APPC type for this transfer, as follows:

S

SNAX/APC

T

TCP/IP

COMMAND

Specifies the type of transfer.

xcom-send-file

Sends a local file to a remote system.

xcom-send-report

Sends a local report to be printed on a remote printer.

xcom-send-job

Sends a job to a remote system.

xcom-receive-file

Retrieves a file from the remote system.

Note: These examples are written in DDL; the names may vary slightly when the API is written in another language (for example, in C, xcom-send-file is written as xcom_send_file).

CONV-SECURITY

Applies to locally initiated transfers.

Specifies whether the user ID/password pair will be sent in the SNA ATTACH request.

Y

Sends the user ID/password pair in the ATTACH request.

N

User ID/password pair is not sent in the ATTACH request.

IDEST

Indicates the intermediate destination name for indirect transfers.

If this variable is null or unset, a direct connection is attempted to the remote system.

If it contains a value, it is assumed to be the name of an intermediate CA XCOM Data Transport destination that will handle traffic to and from the named remote system.

Default: None

LUNAME

Identifies the local LU name to use during this transmission.

MAX-SNAX-IOSIZE

Specifies the size of the buffer passed between CA XCOM Data Transport for HP NonStop and SNAX/APC.

RLUNAME

Identifies the name of the remote LU as configured in APPC.

Default: XCOMAPPL

VERSION

Locally initiated transfers only.

Indicates whether the request is a Version 1 or Version 2 transfer.

1

Version 1

2

Version 2

Default: 2

XDIR

Specifies the default volume and subvolume for all files that are read and written by CA XCOM Data Transport for HP NonStop except the XCOMCNF, XCOMHIST, XCOMPWF, and CKPTFIL files.

Note: If a transfer is initiated remotely and XDIR is not specified, CA XCOM Data Transport uses the default volume of the user ID that the remote system sends.

Range: Up to 256 characters.

Default: None

XLOGFILE

For locally initiated CA XCOM Data Transport transfers only.

Provides a file name for the log file for locally initiated transfers.

Range: Up to 250 characters.

Default: XCOMLOG

XMODE

Specifies the mode name that CA XCOM Data Transport for HP NonStop will use during this transmission.

Range: Up to eight characters

Default: XCOMMODE

API Send and Retrieve File Parameters

The following sections describe the send and retrieve file parameters for the API.

ASCEBC

Indicates which file to use for ASCII to EBCDIC conversion.

CACHE-BUF

Lets you specify where you want to write records.

Y

Writes records to cache instead of directly to disk.

N

Writes records to disk.

CHECKPOINT-COUNT

Indicates the number of records between checkpoints.

CHECKPOINT-FILE

Indicates the file to which the checkpoint requests are written.

COMPRESSION

Indicates if compression is to be used during the transmission of a file. Compressing data may decrease transmission time.

N

No compression

Y

Simple compression

0

RLE compression

1

Compact

2

LCOMPACT

S

LZSMALL

M

LZMEDIUM

L

LZLARGE

H

HUFFMAN

W

LZRW3

Z

ZLIB

Note: When sending text files to an IBM AS/400 or System/36, set COMPRESS to Y to overcome the problem of zero-length lines. Using compression guarantees that all lines will have at least one character to satisfy the LU 6.2 read on the receiving end.

EBCASC

Indicates which file to use for EBCDIC to ASCII conversion.

FILEACTION

Indicates how the remote system should process the transferred data.

C

Create a new file on the receiving system.

A

Append this data to an existing file on the receiving system.

R

Replace the contents of an existing file on the receiving system.

HISTORY-FILE

Indicates the file to which the history records are written.

NULLFILL

Indicates whether outgoing EBCDIC text records should be filled with null characters at the end of the record.

N

Do not use null characters at the end of the record.

Y

Use null characters at the end of the record.

Default: N

PACK

Speeds data transmission by packing records into 2KB blocks before sending to the remote system.

Y

Pack the data before transmission.

N

Do not pack the data.

REQUEST-NO

A unique ID number that the system assigns to each transfer.

RFILE

Specifies the name of the file on the remote system.

If you are creating the file, make sure your designated file name follows the remote system's file naming conventions. The local CA XCOM Data Transport for HP NonStop system does not validate this name; the remote I/O system will determine if the name is valid.

If the file name has a backslash (****) in the name, then enter *two* backslashes together, instead of one: enter *drive*:*****dirname******filename.ext* instead of *drive*:*****dirname******filename.ext*.

Example:

Enter c:****mydocs****list2.doc instead of c:****mydocs****list2.doc.

RESTART-FLAG

Indicates if a transfer is a restart request. RESTART-FLAG can be used to force a restart.

Y

This is a restart request.

N

This is not a restart request.

For more information about restarting failed transfers, see the chapter "Operation and Control."

SIO

Lets you read or write Tandem EDIT files to/from disk using TANDEM SIO (Sequential Input/Output) instead of ANSI C procedures.

Y

Use TANDEM SIO procedures.

N

Use ANSI C procedures.

Note: SIO has been deprecated and will be set to SIO=N and CACHEBUF=Y if specified

TRANSFER-ID

A non-unique ID number that the user assigns to each transfer.

XBUFFSIZE

Specifies the buffer size for a single record. Set this to the maximum record size for the transfer.

Note: For HP NonStop records, the maximum record size is 4096.

Default: 4096

XFILE

Specifies the name of the file on the local system. All Tandem file naming standards apply.

API Disk File Creation Parameters

The following sections describe the disk file creation parameters for the API.

CARRIAGEFLAG

Controls the treatment of text files.

If CARRIAGEFLAG=Y and CODEFLAG=ASCII or EBCDIC, new line characters are added to incoming records and removed from outgoing records.

CODEFLAG

Identifies the type of data being transferred so that CA XCOM Data Transport for HP NonStop knows if it should translate the data.

B

A binary file such as an executable file is being transferred. No translation required.

A

An ASCII file is being transferred. No translation required.

E

If HP NonStop is sending the file, it translates it into EBCDIC; if HP NonStop is receiving the file, it translates it back to ASCII.

Default: EBCDIC

Note: CA XCOM Data Transport for HP NonStop translates every byte in the file. If you have mixed characters and binary data, the file will be corrupted if you specify EBCDIC.

GUARDIANFILETYPE

Indicates the type of ENSCRIBE file to create for a locally initiated transfer.

ED

Edit

EN

Entry sequence

RE

Relative

UN

Unstructured

PRIALLOC

Specifies the primary size extent for creating local and remote files.

Default: 2

RECFM

Specifies the record format for the file being created. This corresponds to the JCL RECFM subparameter.

F

Fixed Unblocked. All records have the same length.

FB

Fixed Blocked. A fixed record length with multiple records per block.

VB

Variable Blocked. Variable length records with multiple records per block.

U

Undefined. The records are of undefined length.

Default: VB

SECALLOC

Specifies a secondary size extent for creating local and remote files.

Default: 4

API IBM Mainframe File Creation Parameters

The following sections describe IBM mainframe file creation parameters for the API.

ALLOC-UNIT

Used only when creating mainframe files.

Specifies the size of the allocation unit if the remote is an IBM mainframe. The actual byte count of each type will vary, depending on the storage device.

B

Blocks

C

Cylinders

T

Tracks

BLKSIZE

Specifies the physical block size of a file. The range depends on record length.

For a variable record format

BLKSIZE = LRECL + 4

For a fixed or fixed blocked record format

BLKSIZE = a multiple of LRECL

For an undefined record format

BLKSIZE > largest record length

Note: If you create a structured file on the HP NonStop system, it must be a valid HP NonStop block size. CA XCOM Data Transport computes an appropriate value.

Range: Up to five characters

Default: 4096

COMPRESS_PDS

Applies to z/OS only.

COMPRESS_PDS is the parameter that causes the actual PDS compression to happen. If your CA XCOM Data Transport z/OS administrator has enabled the programmatic PDS compression feature in a CA XCOM Data Transport region, you can use the COMPRESS_PDS option to control if and when output PDS data sets get compressed as part of the transfer.

Note: COMPRESS_PDS applies only to PDS data sets that will be, or have been, opened for output as the target of a CA XCOM Data Transport transfer.

NONE

Suppresses the compression of an output PDS data set as part of a CA XCOM Data Transport transfer.

BEFORE

Causes an output PDS data set to be compressed before the transfer of user data begins.

AFTER

Causes an output PDS data set to be compressed after the transfer of user data has completed.

BOTH

Causes an output PDS data set to be compressed both before and after the transfer of user data.

Default: NONE

CREATEDELETE

Applies to z/OS only.

CREATEDELETE specifies whether an existing z/OS data set should be deleted and a new data set allocated at the start of a FILE_OPTION=CREATE transfer.

YES

If FILE_OPTION=CREATE and the data set exists, then the z/OS data set is deleted and a new data set is allocated at the start of the transfer.

NO

If FILE_OPTION=CREATE and the z/OS data set exists, then the transfer fails with a catalog/file error.

Default: NO

Notes:

- Specifying CREATEDELETE=YES causes the attributes of the existing data set to be lost; the new data set is allocated with the attributes specified in the transfer.
- CREATEDELETE applies only if the target data set is a sequential data set or an entire PDS/PDSE. CREATEDELETE is ignored for other types of data sets (such as PDS members, PDSE members, VSAM, and USS files).
- CREATEDELETE does not apply to relative GDGs unless the data set is specified using the fully qualified G*xxxx*V*xx* name.
- The use of CREATEDELETE=YES must be allowed by your site's CA XCOM Data Transport administrator for z/OS through the default table (XCOMDFLT) or destination member (XCOMCNTL).

LRECL

Specifies the actual or maximum length in bytes of a logical record. This corresponds to the JCL LRECL subparameter.

For a variable blocked format

LRECL should equal the maximum record length.

For a fixed or fixed blocked format

LRECL should equal the constant record length.

Range: Up to five characters

Default: 0, except in the following cases:

- If GUARDIAN_FILE_TYPE=EDIT or UNSTRUCTURED, the default is 239, or 243 for variable blocked.
- If GUARDIAN_FILE_TYPE=RELATIVE or ENTRYSEQ, the default is taken from the record length parameter in the transferred file.

RECFM

Specifies the record format for the file being created. This corresponds to the JCL RECFM subparameter.

F

Fixed Unblocked. All records have the same length.

FB

Fixed Blocked. A fixed record length with multiple records per block.

VB

Variable Blocked. Variable length records with multiple records per block.

U

Undefined. The records are of undefined length.

Default: VB

SYSTEM-USER-DATA

System-dependent user data included with each transfer.

TRANSFER-USER-DATA

Indicates any user-specified information for each transfer that can be passed to the remote system. This information is written in the history record.

VOLUME

IBM Mainframe File Creation parameter

Specifies the volume on which to create the file.

Range: Up to six characters

Default: None

XUNIT

Specifies the unit on which to create the file. This parameter is ignored when files are created on the Tandem.

SMS Information

DATACLAS

Specifies the name of the data class to use when allocating a new SMS-managed data set.

Note: This parameter applies only to mainframe SMS data sets.

Range: One to eight characters

Default: None

DSNTYPE

Specifies the data set definition.

Note: This parameter applies only to mainframe SMS data sets.

LIBRARY

Defines a PDSE.

PDS

Defines a partitioned data set.

Note: These values are IBM standards for SMS processing.

Range: One to eight characters

Default: None

MGMTCLAS

Specifies the name of the management class to use when allocating a new SMS-managed data set.

Note: This parameter applies only to mainframe SMS data sets.

Range: One to eight characters

Default: None

STORCLAS

Specifies the name of the storage class for a new SMS-managed data set.

Note: This parameter applies only to mainframe SMS data sets.

Range: One to eight characters

Default: None

Tape Information

DEN

Specifies the density to be used in creating a tape on the remote system. Valid values are the same as those for the DEN parameter in JCL.

Range: 1 to 4

Default: None

EXPDT

Specifies an expiration date for the tape data set in terms of a two-digit designation for the year and a three-digit designation for the day of the year.

Example:

In the expiration date 11021, 11 is the year (namely, 2011) and 021 is the 21st day of that year, when the tape data set expires.

Format: *yyddd*

Default: None

Note: EXPDT and RETPD are mutually exclusive; specify one or the other.

LABELNUM

Indicates the sequence number of the data set on the tape.

Sequence number (0001 to 9999)

This value identifies the sequence number of a data set on tape.

Example:

LABELNUM=2

This specification refers to the second data set on the tape.

Default: 0001

RETPD

Specifies the number of days (1 to 9999) that the tape data set being created is to be retained.

Range: 1 to 9999

Default: None

Note: RETPD and EXPDT are mutually exclusive; specify one or the other.

TAPE

Indicates to the remote system whether the volume is a tape volume or a disk file.

YES

Indicates a tape volume and that mounts are allowed when performing dynamic allocation.

NO

Indicates that the transfer is to a disk file.

Default: None

TAPE_LABEL

Indicates the type of label associated with a tape data set. The following table lists the valid values for this parameter.

Processing type (AL, AUL, BLP, LTM, NL, NSL, SL, SUL)

Represents the type of processing to be applied to data sets on tape.

Note: CA XCOM Data Transport for z/OS supports only standard label tapes.

Example:

LABEL=BLP

The type of processing to be applied to this data set is BLP.

Default: AL

TAPEDISP

Specifies the disposition value for MVS tape data sets.

1

New

2

Old

3

Mod

Default: 1

UNITCT

Specifies the number of units to be allocated on the remote system. This is a tape parameter and is used when the partner is an IBM mainframe.

Range: 1 to 20

Default: None

VOLCT

Specifies the maximum number of volumes to be used in processing a multi-volume output tape data set on the remote system.

Range: 1 to 255

Default: None

VOLSQ

Specifies the sequence number of the first volume of a multi-volume remote data set to be used.

Range: 1 to 255

Default: None

API Send Report Parameters

The following sections describe send report parameters for the API.

CARRIAGECONTROL

Indicates which carriage control characters are used in the print job.

A

ASA control codes in column 1

M

IBM Machine Characters (for z/OS only)

Default: No carriage control codes are used.

CHARS

Specifies which font to use for reports sent to the z/OS system.

COPIES

Indicates the number of copies to be printed.

Range: 0 to 999

Default: 1

DISPOSITION

Indicates the disposition of the printed file after printing is completed. Whether this parameter is meaningful is system dependent.

D

Delete after printing.

K

Keep after printing.

H

Hold after printing.

EOL-CLASSES

This characters string contains print classes for which an ASCII NL character will be appended to each record. This is determined by the remote system.

FCB

For reports sent to a z/OS mainframe, FCB specifies the forms control block (FCB) JCL parameter that defines print density, lines per page, and so on. This parameter is ignored for report printing on the HP NonStop system.

FORM

Specifies which forms the printed output should use.

Because CA XCOM Data Transport for HP NonStop places the print job in the remote system's print queue, the print control functions depend on the remote system. Before sending a report, you must verify that the type of form you are requesting is available at the remote site.

Note: When sending a report to an OpenVMS system, leave the FORM parameter blank unless you are certain that you are entering a valid form type value. OpenVMS interprets a blank to mean that no special form is being requested.

HOLDFLAG

Indicates the transferred report file's printing status on the remote system.

Y

Place the file on HOLD on the remote system.

N

Prepare the file for immediate printing.

REPORTTITLE

Specifies the report title that will be printed on the job separator. This varies according to the type of remote (receiving) system, as follows:

System/38

CPF assumes REPORTTITLE to be the printer file name.

MVS

A non-blank REPORTTITLE value generates a separator (banner) page.

OpenVMS

This title is printed with the report.

SUN3 or SUN4

The REPORTTITLE field is passed to the local print spooler as a title field.

Other systems

The REPORTTITLE field is generally used only as a descriptive comment and is not printed as part of the report.

SPOOLFLAG

Indicates to the remote system whether it should "spool" the report received from the local system. HP NonStop will send all reports that it receives to a spooler.

Y

Spool the report received from the local system.

N

Do not spool the report.

XCLASS

Indicates the print class for the print job.

If printing on HP NonStop, this parameter is ignored.

If the remote system is a z/OS system, then XCLASS designates the JES SYSOUT class. In this case, to print the report through SYSOUT=B, enter B.

XDESTINATION

Indicates the destination on the remote system for the print job. If unspecified, the remote system will send the print job to the system's default printer.

For report printing on the Tandem system, the remote system should specify the destination as #loc-name. The spool collector name is specified in the parameter SPOOLCOLLECTOR.

API Job Information Parameters

The following sections describe the job information parameters for the API.

JOBNAME

The name of the job as determined by the remote system.

JOBNUMBER

The number of the job as determined by the remote system.

API Store-and-Forward Parameters

The following sections describe the store-and-forward parameters for the API.

IDEST

Indicates the intermediate destination name for indirect transfers.

If this variable is null or unset, a direct connection is attempted to the remote system.

If it contains a value, it is assumed to be the name of an intermediate CA XCOM Data Transport destination that will handle traffic to and from the named remote system.

Default: None

RLUNAME

Identifies the name of the remote LU as configured in APPC.

Default: XCOMAPPL

API Notification Parameters

The following sections describe the notification parameters for the API.

LCLNTFYL

Specifies the local user notification level.

ALL

NOTIFY on transfer completion.

WARN

NOTIFY only if the transfer received a warning or error.

ERROR

NOTIFY only if the transfer received an error.

Default: ALL

LOCALNOTIFY

Indicates the name of the user to notify on the local system when CA XCOM Data Transport for HP NonStop has completed the transfer.

NOTIFY

Specifies the notification flag on the remote system.

T

TSO user notification.

W

Write to log only.

C

CICS user notification.

L

Logical unit notification.

V

VM/CMS user notification.

N

No user notification.

Note: This parameter is associated with the WHO parameter.

RMTNTFYL

Specifies the remote user notification level when sending data to a remote system.

ALL

NOTIFY on transfer completion.

WARN

NOTIFY only if the transfer received a warning or error.

ERROR

NOTIFY only if the transfer received an error.

Default: ALL

WHO

Indicates the name of the user to notify on the remote system when CA XCOM Data Transport for HP NonStop has completed its procedure.

If the remote system is a z/OS system, CA XCOM Data Transport for HP NonStop uses the value of WHO to determine the type of notification to deliver. If the remote system is a HP NonStop system, the user will receive a mail message.

TSO-NOTIFY

The notify name for the TSO account as determined by the remote system.

API Spooling Parameters

The following sections describe the spooling parameters for the API.

Note: Spooling parameters are used for local printing only. If the remote user does not specify printing parameters, CA XCOM Data Transport for HP NonStop uses the spooling parameter settings listed below as defaults.

COPIES

Indicates the number of copies to be printed.

Range: 0 to 999

Default: 1

DISPOSITION

Indicates the disposition of the printed file after printing is completed. Whether this parameter is meaningful is system dependent.

D

Delete after printing.

K

Keep after printing.

H

Hold after printing.

EOL-CLASSES

This characters string contains print classes for which an ASCII NL character will be appended to each record. This is determined by the remote system.

FORM

Specifies which forms the printed output should use.

Because CA XCOM Data Transport for HP NonStop places the print job in the remote system's print queue, the print control functions depend on the remote system. Before sending a report, you must verify that the type of form you are requesting is available at the remote site.

Note: When sending a report to an OpenVMS system, leave the FORM parameter blank unless you are certain that you are entering a valid form type value. OpenVMS interprets a blank to mean that no special form is being requested.

HOLDFLAG

Indicates the transferred report file's printing status on the remote system.

Y

Place the file on HOLD on the remote system.

N

Prepare the file for immediate printing.

REPORTTITLE

Specifies the report title that will be printed on the job separator. This varies according to the type of remote (receiving) system, as follows:

System/38

CPF assumes REPORTTITLE to be the printer file name.

MVS

A non-blank REPORTTITLE value generates a separator (banner) page.

OpenVMS

This title is printed with the report.

SUN3 or SUN4

The REPORTTITLE field is passed to the local print spooler as a title field.

Other systems

The REPORTTITLE field is generally used only as a descriptive comment and is not printed as part of the report.

SPOOLCOLLECTOR

Identifies the default location for spooled reports and jobs received from remote systems.

Default: $S.#XCOMJOB

SPOOLFLAG

Indicates to the remote system whether it should spool the report received from the local system. Tandem will send all reports that it receives to a spooler.

API Security Parameters

The following sections describe the security parameters for the API.

CONV-SECURITY

Applies to locally initiated transfers.

Specifies whether the user ID/password pair will be sent in the SNA ATTACH request.

Y

Sends the user ID/password in the ATTACH request.

N

User ID/password pair is not sent in the ATTACH request.

DOMAIN

The Windows domain name for use in authenticating the user ID and password when accessing a Windows based machine that has sharable disks and drives that belong to that domain. This allows users to access these sharable drives without having to have a local user ID or password defined to the machine.

Range: 1 to 15 characters

Default: None

PASSWORD

Indicates the password for use with the remote system's file security scheme.

PASSWORD-FILE

The name of the CA XCOM Data Transport for HP NonStop security file.

REMOTEUSER

Identifies the remote user ID for use with the file security scheme.

API Gateway Parameters

This section describes CA XCOM Gateway parameters used by the API in CA XCOM Data Transport for HP NonStop.

GATEWAYGUID

Identifies the remote file as a CA XCOM Gateway file and specifies the CA XCOM Gateway GUID. The CA XCOM Gateway GUID is a unique value that identifies each CA XCOM Gateway file. The keyword ANY can be used to identify the remote file as a CA XCOM Gateway file when the CA XCOM Gateway GUID is not known.

Range: 0 to 36 characters

Default: None (the remote file is not a CA XCOM Gateway file)

API OpenSSL Parameters

CONFIGFILE

Specifies the name of the SSL configuration file. Use the following format:

vol.subvol.filename

Range: Up to 27 characters.

Default: XCSSLCNF

Note: The value for XDIR will be used if specified.

XCOM-SHOW-CIPHER

Specifies whether to display encryption algorithms in the CA XCOM Data Transport queue detailed information, which is used for transfers.

NO

Do not display encryption algorithms in the queue detail information.

YES

Display encryption algorithms in the queue detail information.

Default: NO

XCOMFULLSSL

Specifies whether to use an OpenSSL socket or non-OpenSSL socket for transfers.

YES

Performs a secure transfer. The transfer uses an OpenSSL socket and must to connect to an SSL listener on the remote partner.

NO

Performs a non-secure transfer. The transfer uses a non-OpenSSL socket.

Default: NO

API TAL Transfer Structure (XAPITAL)

The following shows the TAL transfer structure XAPITAL supplied on the distribution media:

```
?Section API^TRANSFER
?PAGE
!*****************************************************************************
!
!    This file was generated from apiddl
!
!*****************************************************************************
STRUCT       API^TRANSFER^DEF (*);
   BEGIN
   INT          APIVERSION;
   STRUCT       CONFIGFILE;
      BEGIN STRING BYTE [0:35]; END;
   INT          WAIT;
   STRUCT       APPCOPENNAME;
      BEGIN STRING BYTE [0:15]; END;
   STRUCT       APPCPROCESSNAME;
      BEGIN STRING BYTE [0:15]; END;
   STRING       APPCTYPE;
   FILLER       1;
   INT          COMMAND;
   STRUCT       IDEST;
      BEGIN STRING BYTE [0:127]; END;
   STRUCT       LUNAME;
      BEGIN STRING BYTE [0:17]; END;
   STRUCT       MAX^SNAX^IOSIZE;
      BEGIN STRING BYTE [0:5]; END;
   STRUCT       RLUNAME;
      BEGIN STRING BYTE [0:127]; END;
   INT          VERSION;
   STRUCT       XDIR;
      BEGIN STRING BYTE [0:24]; END;
   STRUCT       XLOGFILE;
      BEGIN STRING BYTE [0:35]; END;
   STRUCT       XMODE;
      BEGIN STRING BYTE [0:8]; END;
   STRUCT       ASCEBC;
      BEGIN STRING BYTE [0:26]; END;
   STRUCT       BULKIO;
      BEGIN STRING BYTE [0:1]; END;
   STRING       CACHE^BUF;
   STRUCT       CHECKPOINT^COUNT;
      BEGIN STRING BYTE [0:4]; END;
   STRUCT       CHECKPOINT^FILE;
      BEGIN STRING BYTE [0:26]; END;
   STRING       COMPRESSION;
   STRUCT       EBCASC;
      BEGIN STRING BYTE [0:26]; END;
   STRING       FILEACTION;
   STRUCT       HISTORY^FILE;
```

```
   BEGIN STRING BYTE [0:26]; END;
STRUCT       IPC^PNAME;
   BEGIN STRING BYTE [0:24]; END;
STRUCT       IPC^FNAME;
   BEGIN STRING BYTE [0:150]; END;
STRING       NULLFILL;
STRING       PACK;
STRUCT       REQUEST^NO;
   BEGIN STRING BYTE [0:6]; END;
STRUCT       RFILE;
   BEGIN STRING BYTE [0:254]; END;
STRING       RESTART^FLAG;
STRING       SIO;
STRUCT       TRANSFER^ID;
   BEGIN STRING BYTE [0:10]; END;
STRUCT       XBUFFSIZE;
   BEGIN STRING BYTE [0:6]; END;
STRUCT       XFILE;
   BEGIN STRING BYTE [0:254]; END;
STRING       CARRIAGEFLAG;
STRING       CODEFLAG;
STRUCT       GUARDIANFILETYPE;
   BEGIN STRING BYTE [0:1]; END;
STRUCT       PRIALLOC;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       RECFM;
   BEGIN STRING BYTE [0:3]; END;
STRUCT       SECALLOC;
   BEGIN STRING BYTE [0:5]; END;
STRING       ALLOC^UNIT;
STRUCT       BLKSIZE;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       LRECL;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       SYSTEM^USER^DATA;
   BEGIN STRING BYTE [0:10]; END;
STRUCT       TRANSFER^USER^DATA;
   BEGIN STRING BYTE [0:10]; END;
STRUCT       VOLUME;
   BEGIN STRING BYTE [0:10]; END;
STRUCT       XUNIT;
   BEGIN STRING BYTE [0:10]; END;
STRING       CARRIAGECONTROL;
STRUCT       CHARS;
   BEGIN STRING BYTE [0:4]; END;
STRUCT       COPIES;
   BEGIN STRING BYTE [0:3]; END;
STRING       DISPOSITION;
STRUCT       EOL^CLASSES;
```

```
   BEGIN STRING BYTE [0:127]; END;
STRUCT       FCB;
   BEGIN STRING BYTE [0:4]; END;
STRUCT       FORM;
   BEGIN STRING BYTE [0:10]; END;
STRING       HOLDFLAG;
STRUCT       REPORTTITLE;
   BEGIN STRING BYTE [0:21]; END;
STRING       SPOOLFLAG;
STRING       XCLASS;
STRUCT       XDESTINATION;
   BEGIN STRING BYTE [0:21]; END;
STRUCT       JOBNAME;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       JOBNUMBER;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       UIC;
   BEGIN STRING BYTE [0:9]; END;
STRUCT       XWHEN;
   BEGIN STRING BYTE [0:14]; END;
STRUCT       LOCALNOTIFY;
   BEGIN STRING BYTE [0:64]; END;
STRING       NOTIFY;
STRUCT       TSO^NOTIFY;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       WHO;
   BEGIN STRING BYTE [0:12]; END;
STRUCT       SPOOLCOLLECTOR;
   BEGIN STRING BYTE [0:24]; END;
STRING       CONV^SECURITY;
STRUCT       PASSWORD;
   BEGIN STRING BYTE [0:31]; END;
STRUCT       PASSWORD^FILE;
   BEGIN STRING BYTE [0:26]; END;
STRUCT       REMOTEUSER;
   BEGIN STRING BYTE [0:12]; END;

STRUCT       START^DATE;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       START^TIME;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       RETRY^TIME;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       RETRIES;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       STORCLS;
    BEGIN STRING BYTE [0:8]; END;
STRUCT       DATACLS;
    BEGIN STRING BYTE [0:8]; END;
```

```
   STRUCT       MGTCLAS;
       BEGIN STRING BYTE [0:8]; END;
   STRUCT       DSNTYPE;
       BEGIN STRING BYTE [0:8]; END;
   STRING       EXPDATE^FLAG;         !* future
   STRUCT       TAPE^LABEL;
       BEGIN STRING BYTE [0:3]; END;
   STRING       TAPE;
   STRING       HFS^FLAG;             !* future
   STRUCT       XCOMFULLSSL;
       BEGIN STRING BYTE [0:3]; END;
   STRING       XCOMSHOWCIPHER;       !* future
   STRING       DEN;
   STRUCT       EXPDT;
       BEGIN STRING BYTE [0:5]; END;
   STRUCT       RETPD;
       BEGIN STRING BYTE [0:4]; END;
   STRUCT       UNITCT;
       BEGIN STRING BYTE [0:2]; END;
   STRUCT       VOLCT;
       BEGIN STRING BYTE [0:3]; END;
   STRUCT       VOLSQ;
       BEGIN STRING BYTE [0:3]; END;
   STRUCT       LABELNUM;
       BEGIN STRING BYTE [0:4]; END;
   STRING       TAPEDISP;
   STRUCT       CODETABL;
       BEGIN STRING BYTE [0:3]; END;
   STRUCT       SECLABEL;
       BEGIN STRING BYTE [0:8]; END;
   STRING       LCLNTFYL;
   STRING       RMTNTFYL;
   STRUCT       CONFIGSSL;
       BEGIN STRING BYTE [0:254]; END;
   STRING       TRUSTED;
   STRUCT       DOMAIN;
        BEGIN STRING BYTE [0:15]; END;
   STRUCT       GATEWAYGUID;
         BEGIN STRING BYTE [0:36]; END;
   STRING       CREATEDELETE;
   STRING       COMPRESS^PDS;
   STRUCT       IPPROCESSNAME;
       BEGIN STRING BYTE [0:5]; END;
   STRUCT       PORT;
       BEGIN STRING BYTE [0:5]; END;
   END;
?Section API^COMMANDS
!*****************************************************************************
!
```

```
!    This file was generated from apiddl
!
!*************************************************************************
Literal XCOM^SEND^FILE = 1;
Literal XCOM^SEND^REPORT = 2;
Literal XCOM^SEND^JOB = 3;
Literal XCOM^RECEIVE^FILE = 4;
?Section API^ERROR^CODES
!*************************************************************************
!
!    This file was generated from apiddl
!
!*************************************************************************
!   Error codes returned by XCOM62^API
Literal ERROR^ALLOCATING^SEND^BUFFER = 283;
Literal ERROR^FORKING = 284;
Literal ERROR^CREATING^PIPE = 285;
Literal ERROR^SETTING^LOCAL^USER^ID = 286;
Literal ERROR^SETTING^REMOTE^USER^ID = 287;
Literal ERROR^SYSTEM^FAILED = 288;
Literal ERROR^COMMAND^FAILED = 289;
Literal ERROR^RECEIVING^OVERLAY = 290;
Literal ERROR^SENDING^OVERLAY = 291;
Literal ERROR^SENDING^ERROR = 292;
Literal ERROR^EXPECTING^SEND^STATE = 293;
Literal ERROR^EXPECTING^RECEIVE^STATE = 294;
Literal ERROR^COMMAND^LINE = 295;
Literal ERROR^DEALLOCATING = 296;
Literal ERROR^REQUESTING^HEADR^CONFIRM = 297;
Literal ERROR^ALLOCATE = 298;
Literal ERROR^LOCAL^ATTACH = 299;
Literal ERROR^REMOTE^ATTACH = 300;
Literal ERROR^STARTING^APPC = 301;
Literal ERROR^OPENING^INPUT^FILE = 302;
Literal ERROR^SENDING^HEADER = 303;
Literal ERROR^SENDING^MAXLRECL = 304;
Literal ERROR^RECEIVING^HEADER = 305;
Literal ERROR^INVALID^HEADER = 306;
Literal ERROR^INVALID^PACK^OPTION = 307;
Literal ERROR^READING^INPUT^FILE = 309;
Literal ERROR^RECEIVE^ERROR = 310;
Literal ERROR^SENDING^DATA = 311;
Literal ERROR^CONFIRMING^DATA = 312;
Literal ERROR^NEGATIVE^DATA^CONFIRM = 313;
Literal ERROR^SENDING^TRAILER = 314;
Literal ERROR^NEGATIVE^TRAILER^CONFIRM = 315;
Literal ERROR^RECEIVED^FROM^REMOTE = 316;
Literal ERROR^RECEIVE^FMH^7 = 317;
Literal ERROR^REVERSING^LINE = 318;
```

```
Literal ERROR^CONFIRMING^CHECKPOINT = 319;
Literal ERROR^CONFIRMED^CHECKPOINT = 320;
Literal ERROR^OPENING^CHECKPOINT^FILE = 321;
Literal ERROR^WRITING^CHECKPOINT^FILE = 322;
Literal ERROR^RECV^RECEIVING^HEADER = 401;
Literal ERROR^HEADER^INVALID = 402;
Literal ERROR^OPENING^OUTPUT^FILE = 403;
Literal ERROR^CONFIRMING^HEADER = 404;
Literal ERROR^RECEIVING^MAXLRECL = 405;
Literal ERROR^RECEIVING^FEATURE^RECORD = 406;
Literal ERROR^MAXLRECL^INVALID = 407;
Literal ERROR^FEATURE^RECORD^PROTOCOL = 408;
Literal ERROR^REQUESTING^FEATURE^CNFRM = 409;
Literal ERROR^SENDING^FEATURE^RECORD = 410;
Literal ERROR^CONFIRM^FEATURE^RECORD = 411;
Literal ERROR^RECEIVING^DATA = 412;
Literal ERROR^TRAILER^INVALID = 413;
Literal ERROR^CLOSING^OUTPUT^FILE = 414;
Literal ERROR^RECEIVING^TRAILER = 415;
Literal ERROR^WRITING^OUTPUT^FILE = 416;
Literal ERROR^CONFIRMING^TRAILER = 417;
Literal ERROR^INTERRUPT^RECEIVED = 418;
Literal ERROR^USER^NOT^FOUND = 419;
Literal ERROR^USER^ID^OUT^OF^RANGE = 420;
Literal ERROR^GROUP^ID^OUT^OF^RANGE = 421;
Literal ERROR^LOGIN^INCORRECT = 422;
Literal ERROR^RECEIVE^FILE^DESCRIPTOR = 423;
Literal ERROR^SENDING^FILE^DESCRIPTOR = 424;
Literal ERROR^REPOSITIONING^FILE = 425;
Literal ERROR^RESTART^CNTS^DONT^MATCH = 426;
Literal ERROR^STARTING^TP = 427;
Literal ERROR^NO^SESSIONS^AVAILABLE = 435;
Literal ERROR^TP^ABENDED = 436;
Literal ERROR^INPUT^BUFFER^TOO^SMALL = 437;
Literal ERROR^TRUNCATION^NOT^ALLOWED = 438;
Literal ERROR^MAXLRECL^TOO^BIG = 440;
Literal ERROR^REQNOS^DONT^MATCH = 441;
Literal ERROR^FILENAMES^DONT^MATCH = 442;
Literal ERROR^GROUPNAMES^DONT^MATCH = 443;
Literal ERROR^INVALID^PACKING^TYPE = 448;
Literal ERROR^IN^COMPRESSION^TYPE = 449;
Literal ERROR^MEMORY^ALLOCATION = 468;
Literal ERROR^FILE^ALREADY^EXISTS = 470;
Literal ERROR^IPC^PARAMS^DONT^MATCH = 501;
Literal ERROR^IPC^PROGRAM^NOT^STARTED = 502;
Literal ERROR^IPC^PROCESS^NOT^EXIST = 503;
Literal ERROR^IPC^PROCESS^NAME^ILLEGAL = 504;
Literal ERROR^IPC^PROGRAM^NAME^ILLEGAL = 505;
Literal ERROR^IPC^PROCESS^ERROR = 506;
```

```
Literal ERROR^IPC^DATA^REC^LG^TOO^BIG = 507;
Literal ERROR^MISSING^PARAMETER = 519;
Literal ERROR^API^INCORRECT^VERSION = 520;
```

API TAL Sample Program (APIEXS)

The CA XCOM Data Transport for HP NonStop distribution media include a sample file of a TAL program, as shown below. This program uses CA XCOM Data Transport for HP NonStop to send the local file $XATA3.TEST.TESTFILE to a remote system.

Observe the following rules when using an API program:

- All file names must be in external format.
- When moving appropriate values to structure fields, you must add a null [0] to the end of each two-byte variable field.
- Because the GUARDIANFILETYPE field is a single-byte field (though it appears as a two-byte field), it doesn't need a null at the end of the data. (See Rule 2.)
- The TRANSFERWAIT field offers two options:
  - If set to 0, the program will not wait and the transfer will be queued.
  - If set to a non-zero value, the scheduled transfer does one of the following:
- Completes code.
- Does not complete and returns an appropriate error code.

Note: The supplied file APIEXS can be used to compile this program.

```
?SYMBOLS, INSPECT, SAVEABEND, NOCODE
?RELOCATE
?ERRORS 15

-- ************************************************************************
--
--   XCOM API Example Program.
--
--   N.B. - The API requires
--     ADD DEFINE =xcom62-program, class map, file <xcom program file name>
--
-- ************************************************************************

NAME XCOM^EXAMPLE^PROGRAM;

BLOCK XCOM^EXAMPLE^GLOBALS;
?NOLIST, SOURCE scidict2.xapital      ! TAL definition of XCOM api structure
?SEARCH BULKIO3.apio                  ! XCOM api object code
?LIST

LITERAL
    xcom^true = -1,
    xcom^false = 0;

DEFINE
```

```
    NULL = [0]#;

END BLOCK ; ! XCOM^EXAMPLE^GLOBALS
?SOURCE sci.apideft(xcom62^api) ! External declaration of XCOM62^API
?SOURCE $system.system.extdecs0(AWAITIOX, DELAY, FILEINFO)
PROC calculate^the^mandelbrot^set;
     EXTERNAL;
?PAGE "XCOM^EXAMPLE"
PROC XCOM^EXAMPLE MAIN;

BEGIN

INT
        return^code,
        num^bytes,
        error,
       .EXT pid[0:3],
       .EXT xcom^file^number[0:0],
       .EXT int^ptr
        ;

STRUCT
       .EXT transfer (api^transfer^def)
        ;



-- ************************************************************************
-- Fill the structure to be passed to XCOM62^API with NULLs.
-- ************************************************************************

@int^ptr := @transfer;
int^ptr[0] := 0;
int^ptr[1] ':=' int^ptr[0] FOR (($LEN(transfer) + 1) / 2) - 1;


-- ************************************************************************
-- Move appropriate values to structure fields
-- ************************************************************************

transfer.version := 2;
transfer.apiversion := 3;
transfer.command := xcom^send^file;       ! send a file to remote system !
transfer.wait := 0;                       ! wait for transfer to complete !
transfer.configfile ':=' ["$clx12.sci.rosecnf", 0]; ! name of config file!

transfer.luname ':=' ["LU338B00", 0];     ! name of local LU !
transfer.rluname ':=' ["XCOMQA", 0];      ! name of remote LU !
transfer.idest ':=' NULL;                 ! intermediate destination !
```

```
transfer.xmode ':=' ["XCOMMODE", 0];      ! mode used by XCOM !
transfer.nullfill ':=' ["N"];             ! null fill flag for text files !
transfer.localnotify ':=' ["sci.manager", 0]; ! name of user to notify !
transfer.xfile ':=' ["$clx12.sci.xdefsh", 0]; ! name of local file !
transfer.rfile ':=' ["DA1RL04.APITEST", 0];   ! name of remote file !
transfer.xlogfile ':=' ["sci.xcomlog", 0];! name of xcom log file !
transfer.compression ':=' ["Y"];          ! compression     !
transfer.notify ':=' ["Y"];               ! notify flag !
transfer.who ':=' ["DA1RL04", 0];         ! who to notify on remote system !
transfer.remoteuser ':=' ["DA1RL04", 0];  ! remote user id !
transfer.fileaction ':=' ["C"];           ! file action !
transfer.carriageflag ':=' ["Y"];         ! carriage return flag !
transfer.codeflag ':=' ["E"];             ! code flag !
transfer.password ':=' ["X", 0];          ! remote user password !
transfer.appcprocessname ':=' ["$icel", 0];    ! Snax appc process name !
transfer.appcopenname ':='["#XCOM338", 0];     ! additional qualifiers for ope
transfer.appctype ':=' ["I"];             ! use snax appc or ice calls !
transfer.guardianfiletype ':=' ["ED"];    ! Tandem filetype:           !
                                          !        EDit file           !
                                          !        ENtry sequenced file !
                                          !        RElative file       !
                                          !        UNstructured file   !
transfer.spoolcollector ':=' ["$S1", 0];  ! use this collector for reports !
                                          !     and job execution          !
transfer.xdir ':=' ["$CLX12.SCI", 0];     ! default directory              !
! transfer.max_snax_iosize ':=' [0];  !   ! pick this up from config file  !

     ! File creation parameters !
transfer.prialloc ':=' ["00128", 0];      ! primary allocation             !
transfer.secalloc ':=' ["00512", 0];      ! secondary allocation           !
transfer.recfm ':=' ["VB", 0];            ! record format !
transfer.lrecl ':=' ["00128", 0];         ! lrecl !
transfer.blksize ':=' ["02048", 0];       ! physical block size !
transfer.volume ':=' NULL;                ! volume to create file on !
transfer.xunit ':=' NULL;                 ! unit to create file on !

     ! Job Control Information !
transfer.uic ':=' NULL;
transfer.xwhen ':=' NULL;

     ! Report Control Information !
transfer.xclass ':=' NULL;                ! printjob class !
transfer.xdestination ':=' NULL;          ! name of remote printer !
transfer.form ':=' NULL;                  ! name of special forms to used !
transfer.fcb ':=' NULL;                   ! Form Control Block !
transfer.copies ':=' NULL;                ! number of copies to be printed !
transfer.holdflag ':=' NULL;              ! flag for MVS spoolers !
transfer.reporttitle ':=' NULL;           ! report name to be used on separator sheet !
transfer.carriagecontrol ':=' NULL;
```

```
transfer.spoolflag ':=' NULL;
transfer.disposition ':=' NULL;
transfer.eol^classes ':=' NULL;           ! print classes having NL added at end of record !
transfer.jobname ':=' NULL;
transfer.jobnumber ':=' NULL;
transfer.tso^notify ':=' NULL;

-- -1 indicates nowaited i/o to the XCOM process
return^code := XCOM62^API(transfer, pid, -1, xcom^file^number);

WHILE NOT return^code DO
   BEGIN
   -- Do other productive work here. Occasionally check if the tranfer has
   --    completed.
!  CALL calculate^the^mandelbrot^set;
   CALL AWAITIOX(xcom^file^number,, num^bytes,, 0D);
   IF <>  THEN                     -- error 40 means I/O not yet completed
      BEGIN
      CALL FILEINFO(xcom^file^number, error);
      IF error <> 40 THEN return^code := error;    ! bailout
      END
   ELSE
      BEGIN
      @int^ptr := @transfer;
      return^code := int^ptr; -- api puts error codes in first word of struct
      IF return^code = 0 THEN return^code := -1; -- no more looping
      END;
   END;   -- end of WHILE

END;   -- end of proc
```

API C Transfer Structure (XAPIC)

The following file shows the C transfer structure XAPIC supplied on the distribution media:

```
#pragma section api_transfer
/*****************************************************************************/
/*                                                                           */
/*    This file was generated from apiddl                                    */
/*                                                                           */
/*****************************************************************************/
typedef struct
{
   short                        apiversion;
   char                         configfile[36];
   short                        wait;
   char                         appcopenname[16];
   char                         appcprocessname[16];
   char                         appctype;
   short                        command;
   char                         idest[128];
   char                         luname[18];
   char                         max_snax_iosize[6];
   char                         rluname[128];
   short                        version;
   char                         xdir[25];
   char                         xlogfile[36];
   char                         xmode[9];
   char                         ascebc[27];
   char                         bulkio[2];
   char                         cache_buf;
   char                         checkpoint_count[5];
   char                         checkpoint_file[27];
   char                         compression;
   char                         ebcasc[27];
   char                         fileaction;
   char                         history_file[27];
   char                         ipc_pname[25];
   char                         ipc_fname[151];
   char                         nullfill;
   char                         pack;
   char                         request_no[7];
   char                         rfile[255];
   char                         restart_flag;
   char                         sio;
   char                         transfer_id[11];
   char                         xbuffsize[7];
   char                         xfile[255];
   char                         carriageflag;
   char                         codeflag;
   char                         guardianfiletype[2];
   char                         prialloc[6];
   char                         recfm[4];
   char                         secalloc[6];
```

```
   char                          alloc_unit;
   char                          blksize[6];
   char                          lrecl[6];
   char                          system_user_data[11];
   char                          transfer_user_data[11];
   char                          volume[11];
   char                          xunit[11];
   char                          carriagecontrol;
   char                          chars[5];
   char                          copies[4];
   char                          disposition;
   char                          eol_classes[128];
   char                          fcb[5];
   char                          form[11];
   char                          holdflag;
   char                          reporttitle[22];
   char                          spoolflag;
   char                          xclass;
   char                          xdestination[22];
   char                          jobname[9];
   char                          jobnumber[9];
   char                          uic[10];
   char                          xwhen[15];
   char                          localnotify[65];
   char                          notify;
   char                          tso_notify[9];
   char                          who[13];
   char                          spoolcollector[25];
   char                          conv_security;
   char                          password[32];
   char                          password_file[27];
   char                          remoteuser[13];

   char                          start_date[9];
   char                          start_time[9];
   char                          retry_time[6];
   char                          retries[6];

/* volke01 xcomr11 new header parms start */
   char                          storcls[9];
   char                          datacls[9];
   char                          mgtclas[9];
   char                          dsntype[9];
   char                          expdate_flag;      /* future */
   char                          tape_label[4];
   char                          tape;
   char                          hfs_flag;          /* future */
   char                          xcomfullssl[4];
   char                          xcomshowcipher;    /* future */
```

```
   char                          den;
   char                          expdt[6];
   char                          retpd[5];
   char                          unitct[3];
   char                          volct[4];
   char                          volsq[4];
   char                          labelnum[5];
   char                          tapedisp;
   char                          codetabl[4];
   char                          seclabel[9];
   char                          lclntfyl;
   char                          rmtntfyl;
   char                          configssl[255];
   char                          trusted;
   char                          domain[16];
   char                          gatewayguid[37];
   char                          createdelete;
   char                          compress_pds;
   char                          ipprocessname[16];
   char                          port[6];
} api_transfer_def;
#pragma section api_commands
/*****************************************************************************/
/*                                                                           */
/*    This file was generated from apiddl                                    */
/*                                                                           */
/*****************************************************************************/
#define xcom_send_file 1
#define xcom_send_report 2
#define xcom_send_job 3
#define xcom_receive_file 4
#pragma section api_error_codes
/*****************************************************************************/
/*                                                                           */
/*    This file was generated from apiddl                                    */
/*                                                                           */
/*****************************************************************************/
/*   Error codes returned by XCOM62^API                                      */
#define error_allocating_send_buffer 283
#define error_forking 284
#define error_creating_pipe 285
#define error_setting_local_user_id 286
#define error_setting_remote_user_id 287
#define error_system_failed 288
#define error_command_failed 289
#define error_receiving_overlay 290
#define error_sending_overlay 291
#define error_sending_error 292
#define error_expecting_send_state 293
```

```
#define error_expecting_receive_state 294
#define error_command_line 295
#define error_deallocating 296
#define error_requesting_headr_confirm 297
#define error_allocate 298
#define error_local_attach 299
#define error_remote_attach 300
#define error_starting_appc 301
#define error_opening_input_file 302
#define error_sending_header 303
#define error_sending_maxlrecl 304
#define error_receiving_header 305
#define error_invalid_header 306
#define error_invalid_pack_option 307
#define error_reading_input_file 309
#define error_receive_error 310
#define error_sending_data 311
#define error_confirming_data 312
#define error_negative_data_confirm 313
#define error_sending_trailer 314
#define error_negative_trailer_confirm 315
#define error_received_from_remote 316
#define error_receive_fmh_7 317
#define error_reversing_line 318
#define error_confirming_checkpoint 319
#define error_confirmed_checkpoint 320
#define error_opening_checkpoint_file 321
#define error_writing_checkpoint_file 322
#define error_recv_receiving_header 401
#define error_header_invalid 402
#define error_opening_output_file 403
#define error_confirming_header 404
#define error_receiving_maxlrecl 405
#define error_receiving_feature_record 406
#define error_maxlrecl_invalid 407
#define error_feature_record_protocol 408
#define error_requesting_feature_cnfrm 409
#define error_sending_feature_record 410
#define error_confirm_feature_record 411
#define error_receiving_data 412
#define error_trailer_invalid 413
#define error_closing_output_file 414
#define error_receiving_trailer 415
#define error_writing_output_file 416
#define error_confirming_trailer 417
#define error_interrupt_received 418
#define error_user_not_found 419
#define error_user_id_out_of_range 420
#define error_group_id_out_of_range 421
```

```
#define error_login_incorrect 422
#define error_receive_file_descriptor 423
#define error_sending_file_descriptor 424
#define error_repositioning_file 425
#define error_restart_cnts_dont_match 426
#define error_starting_tp 427
#define error_no_sessions_available 435
#define error_tp_abended 436
#define error_input_buffer_too_small 437
#define error_truncation_not_allowed 438
#define error_maxlrecl_too_big 440
#define error_reqnos_dont_match 441
#define error_filenames_dont_match 442
#define error_groupnames_dont_match 443
#define error_invalid_packing_type 448
#define error_in_compression_type 449
#define error_memory_allocation 468
#define error_file_already_exists 470
#define error_ipc_params_dont_match 501
#define error_ipc_program_not_started 502
#define error_ipc_process_not_exist 503
#define error_ipc_process_name_illegal 504
#define error_ipc_program_name_illegal 505
#define error_ipc_process_error 506
#define error_ipc_data_rec_lg_too_big 507
#define error_missing_parameter 519
#define error_api_incorrect_version 520
```

API C Sample Program (APIC)

The CA XCOM Data Transport for HP NonStop distribution media include the C program APIC, which sends a file. A sample of this file is as follows:

```
#pragma NOLIST
#include <tal.h>
#include <errno.h>
#include <fcntlh>
#include <string.h>
#include <stdio.h>
#include "$dsmscm.xctndm11.xapic"
#include <cextdecs(WRITEREADX,WRITEX,OPEN,WRITE,CLOSE,READX)>
#pragma LIST
short XCOM62_API (api_transfer_def *, short *, short, short *, short);
#pragma function XCOM62_API (alias("XCOM62^API"), extensible, tal)


static short pid[12];
static short file_number;
static short fnum;
api_transfer_def  transfer;

char buf[80];
main()
{
int rcv_des,len;
int err,message_tag,sync_id;


int status,i;
rcv_des = open(rcv_file_name, (O_BINARY | O_RDWR | O_SYSMSG),, 2);
transfer.command = xcom_receive_file;

transfer.wait = 0;
strcpy(transfer.configfile,"$dsmscm.xctndm11.icecnf");

strcpy(transfer.luname,"CPICSEND");
strcpy(transfer.rluname,"CPICRECV");
transfer.idest[0] = 0;
strcpy(transfer.xmode,"XCOMMODE");
strcpy(transfer.xfile,"$dsmscm.xctndm11.test");
strcpy(transfer.rfile,"$dsmscm.xctndm11.apifile");

strcpy(transfer.remoteuser,"xcom.xcom");
strcpy(transfer.password,"zxcom");
strcpy(transfer.xlogfile,"xcomlog");
transfer.codeflag = 'E';
strcpy(transfer.guardianfiletype,"VB");
```

```
transfer.fileaction = 'R';
transfer.version = 2;          /* CA XCOM Version 2 protocol     */
transfer.apiversion = 3;       /* CA XCOM for HP NonStop r11 API */

status =  XCOM62_API(&transfer, pid, 0, &file_number, 0);


}
```

Chapter 7: The Interprocess Communications Interface

CA XCOM Data Transport for HP NonStop provides an Interprocess Communications Interface (IPC) that allows logical records to be passed to another process for handling. This section contains additional technical information about the IPC. The following topics are included:

- CA XCOM Data Transport parameter changes
- Initializing communication to a user process
- Sending an open message, and the IPC transfer header reply
- Passing IPC records
- Data flows
- Logic flows
- New error messages
- The API structures

This section contains the following topics:

Using CA XCOM Data Transport Parameters

The following sections contain information about the following parameters and how to use them for both locally and remotely initiated transfers:

- IPC_PNAME
- IPC_FNAME
- BLKSIZE
- XBUFFSIZE
- LRECL

For Locally Initiated Transfers

The following sections describe the parameters used for locally initiated transfers.

Note: Specifying IPC_PNAME and IPC_FNAME in the global XCOMCNF file causes all remotely initiated transfers to go through IPC. IPC_PNAME and IPC_FNAME should only be used in a different XCOMCNF file, which would be referenced by the command line interface.

For remotely initiated transfers, specify the values for these parameters in the remote file field. For example, in the CA XCOM Data Transport z/OS SYSIN01, code the following:

```
FILE=$DSV.QAXCOM.MYFILE*$DSV.QAXCOM.MYAPPL*PARM1,PARM2
```

Note: If the IPC_NO_REMOTE parameter is set to YES, then any IPC information provided by a remote system is ignored.

IPC_PNAME

The process name from which CA XCOM Data Transport reads data or to which it sends data, entered in the following format:

```
<$process><.#qualifier1><.#qualifier2>
```

If an IPC_FNAME is not specified, the IPC process must be running already.

If the IPC process is not running, CA XCOM Data Transport starts one from the IPC_FNAME information.

Note: An IPC_PNAME is required for locally initiated transfers.

Range: 1 to 24 characters, beginning with the character $

Default: None

IPC_FNAME

Specifies the program name (and optional startup parameters specific to your program) that will run if the process specified in the IPC_PNAME does not exist.

The file name is specified in external format. Use a space to delimit the filename from the startup parameters, and the startup parameters from each other, as follows:

```
$CLX01.EXAMPLE.MYAPPL  PARM1 PARM2 PARM3
```

Note: If the full path name is not specified, the XDIR parameter supplies the missing volume and subvolume names.

Range: 1 to 150 characters, beginning with the character $

Default: None

For Remotely Initiated Transfers

Remotely initiated transfers use the IPC_PNAME and IPC_FNAME as described in the following section.

IPC_PNAME or IPC_FNAME

The remote platform appends the information intended for the HP NonStop's IPC_PNAME and IPC_FNAME fields in its remote file name parameter field.

In the remote file name field, use an asterisk to delimit the file name from the appended information, and use commas to delimit the program's startup parameters from each other.

Example:

This example illustrates the correct format for appending information:

```
REMOTE_FILENAME=
$CLX02.XCOM.MYFILE*$MYAP*$CLX01.MYAPPL*PARM1,PARM2,PARM3
```

Note: z/OS treats a blank character as the end of the field.

Existing Parameter Changes

The list below describes the changes for existing CA XCOM Data Transport for HP NonStop parameters:

BLKSIZE

Specifies the physical block size of a file.

If a locally initiated transfer is being sent to another user process, you can enter a block size up to 30720 bytes (30 KB).

If a remotely initiated transfer is being sent to a user process, the remote platform can now specify a block size value up to 30720 bytes (30 KB).

Range: 0 to 30720 bytes

Default: 4096 bytes

XBUFFSIZE

Specifies the buffer size for a single record. Set this to the maximum record size for the transfer.

For Tandem records, the maximum record size is 4096.

Range: 0 to 30720 bytes

Default: 4096 bytes

LRECL

Specifies the actual or maximum length in bytes of a logical record. This corresponds to the JCL RECL subparameter.

For a variable blocked format

LRECL should equal the maximum record length.

For a fixed or fixed block format

LRECL should equal the constant record length.

Range: 0 to 30720 bytes

Default:

- For EDIT or Unstructured files, 239 bytes, or 243 for variable blocked.
- For Relative, Entry Sequence, or Key files, the default is the record length of the transferred file.

Initializing Communication to an HP NonStop Process

After you have entered appropriate values for the IPC parameters, you can establish communication with a user process in the following situations:

- The process does not exist.
- The process exists, and CA XCOM Data Transport for HP NonStop must make itself known to the process.
- The process exists, and it knows CA XCOM Data Transport for HP NonStop exists because it has started CA XCOM Data Transport by using the API.

The following list describes what happens for each of these conditions:

The user process does not exist.

- CA XCOM Data Transport for HP NonStop starts the process.
- CA XCOM Data Transport for HP NonStop opens a channel to the process.
- CA XCOM Data Transport for HP NonStop sends the startup message.
- CA XCOM Data Transport for HP NonStop closes the channel and waits for the process to open CA XCOM Data Transport for HP NonStop.

The user process exists.

- CA XCOM Data Transport for HP NonStop opens a channel to the process.
- CA XCOM Data Transport for HP NonStop sends the startup message.
- CA XCOM Data Transport for HP NonStop closes the channel and waits for the process to open CA XCOM Data Transport for HP NonStop.

The user process exists and has started CA XCOM Data Transport for HP NonStop by using the API.

- CA XCOM Data Transport for HP NonStop waits for the process to open it.

According to the HP NonStop requester/server model, CA XCOM Data Transport for HP NonStop always functions as the server and the process functions as the requester. Because CA XCOM Data Transport for HP NonStop can also initiate communication with the process, this may not always seem to be the case; however, when communication is established, the function of CA XCOM Data Transport for HP NonStop is to REPLY to the WRITEREADs of the process.

The CA XCOM Data Transport-to-User-Process Startup Message

The following example shows the format of the startup message from CA XCOM Data Transport for HP NonStop to the user process:

```
struct startup_msg {
   int msgcode;             /* msgcode = -1                          */
   struct envdefault {
       int volume[4];       /* volume in CA-XCOM XDIR parameter    */
       int subvolume[4];    /* subvolume in CA-XCOM XDIR parameter */
   } deflt;
   struct inf {
        int volume[4];      /* CA-XCOM process name                */
        int subvolume[4];   /* blank                               */
        int  dname[4];      /* blank                               */
    } infile;
     struct outf {
        int volume[4];      /* home terminal name                  */
        int subvolume[4];   /* home terminal name                  */
        int  dname[4];      /* home terminal name                  */
    } outfile;
     string parms[n+7];     /* "CA-XCOM62" PARM1 PARM2 PARM3       */
  } startup;  /*  n is length of parameters in IPC filename        */
```

Note: Although this example is written in C language, you can write this process using any language that HP NonStop supports.

Sending an Open Message/Receiving an IPC Transfer Header Reply

When the user process is running and knows where to find CA XCOM Data Transport for HP NonStop, the process opens a channel to CA XCOM Data Transport for HP NonStop and sends an open message with the following information:

- CA XCOM Data Transport for HP NonStop process name
- User process name
- Block size
- Restart record number

If the CA XCOM Data Transport for HP NonStop process name in the open message doesn't match the CA XCOM Data Transport for HP NonStop process information, or if the user process name doesn't match the startup message destination, the system returns an error 491 in the reply. If everything matches, CA XCOM Data Transport for HP NonStop starts (if locally initiated) or continues (if remotely initiated) its conversation with the remote CA XCOM Data Transport platform.

If the remote platform is ready to send/receive data, CA XCOM Data Transport for HP NonStop sends the user process a reply message in the form of an IPC transfer header. If the remote platform is not ready, CA XCOM Data Transport for HP NonStop sends the user process an IPC record message with a CA XCOM Data Transport error code.

Record Blocking

If you want CA XCOM Data Transport for HP NonStop to send multiple data records in an IPC record, enter a value in the BLKSIZE field to specify the size of the data block. Each data record in the block starts on an even byte boundary and is preceded by a two-byte header that specifies the record size. This record size can't exceed the LRECL value specified by the local or remote CA XCOM Data Transport that initiated the transfer.

If you do not want to use record blocking, set the BLKSIZE parameter to zero.

Note: Data records must not span blocks, and must not exceed the maximum record length specified in the transfer header.

Restarts

The IPC transfer header contains data for initializing the transfer (such as file name, file type, and so on). The RESTART field in the open message specifies one of the following:

- The last record the user process sent or received successfully.
- The record from which it needs to restart.

CA XCOM Data Transport for HP NonStop indicates in the transfer header's RESTART field which data record it is starting from. If the process has no data for the failed transfer, enter a zero in the RESTART field.

For locally initiated transfers, you can accept the restart record (negotiated by the CA XCOM Data Transport platforms) which is returned in the transfer header message. Or, if the RESTART value is zero, CA XCOM Data Transport for HP NonStop begins the transfer all over again.

Note: For remotely initiated transfers, CA XCOM Data Transport ignores the RESTART value.

The User-to-CA XCOM Data Transport Open Message

The format of the open message from CA XCOM Data Transport to the user process is as follows:

```
struct open_msg {
   int msgcode;       /* msgcode = -30        */
   char xcom_pid[6];  /* CA-XCOM process name */
   char user_pid[6];  /* User process name    */
    int blksize;
    long restart;
  } open;
```

The CA XCOM Data Transport-to-User IPC Transfer Header Message

The format of the IPC transfer header message is as follows:

```
struct IPC_header_msg {
   int    ipc_version;      /*  version = 2                                    */
   char transfer_direction; /* 'S' - send records to CA-XCOM, 'R' - read records  */
   char data_type           /* "A" - ASCII,  "B"- binary                       */
   char file_action         /* "C"- Create, "R" - Replace, "A" - Append        */
   char  restart;           /* 'Y' - yes, it's a restart, 'N' - new transfer   */
   char filename[36];       /* filename in external format                     */
   char file_type[2];       /* "UN" - unstructured,  "RE" - relative,  "ED" - edit,
                               "EN" - entry sequence                           */
   char userid[12];         /* userid passed to CA-XCOM                        */
   char password[31];       /* password passed to CA-XCOM                      */
   int lrecl;               /* maximum record size up to 30KB                  */
   int blksize;             /* records packed in blocks up to 30KB             */
   long restart_record;     /* if restart, record number where to start        */
} IPC_header;
```

Note: Although this example is written in C language, you can write this process using any language that HP NonStop supports.

Passing the IPC Records

An IPC record consists of the data that you are transmitting and a header that contains the following information:

- User process
- CA XCOM Data Transport for HP NonStop process
- Status code
- Size of data

If blocking is turned on, the IPC record contains multiple data records. Each data record is preceded by a two-byte record length value that is smaller than the record length in the IPC transfer header.

If CA XCOM Data Transport for HP NonStop is sending a data record, the transfer status is zero. If there is no more data and an end-of-file condition is being sent, the transfer status is 1. All other status values indicate an error condition, and the data part of the record specifies the details of the error.

The IPC Send

In an IPC send (transfer direction in IPC header = S), the user process sends the IPC record, and CA XCOM Data Transport for HP NonStop replies with an IPC record header indicating the status.

If the status is zero, CA XCOM Data Transport for HP NonStop received the record. It does not mean that CA XCOM Data Transport for HP NonStop sent the record successfully to the remote destination.

If the status is not zero, the status value is a CA XCOM Data Transport error number.

Note: An error code of 496 or 497 refers to the record just sent; all other error codes refer to the previous record sent.

The IPC Receive

In an IPC receive (transfer direction in IPC header= R), the user process sends an IPC record with the status of the previous read, and CA XCOM Data Transport for HP NonStop replies with the next IPC record. After CA XCOM Data Transport for HP NonStop sends an IPC record with a status of 1 (end-of-file condition), a final exchange of IPC records with the status set to 1 confirms to both sides that all data has been received by the user process.

The IPC Record Format

The format of the two-way IPC record exchange between CA XCOM Data Transport for HP NonStop and the user process is shown below. Although this example is written in C language, you can write this process using any language that HP NonStop supports.

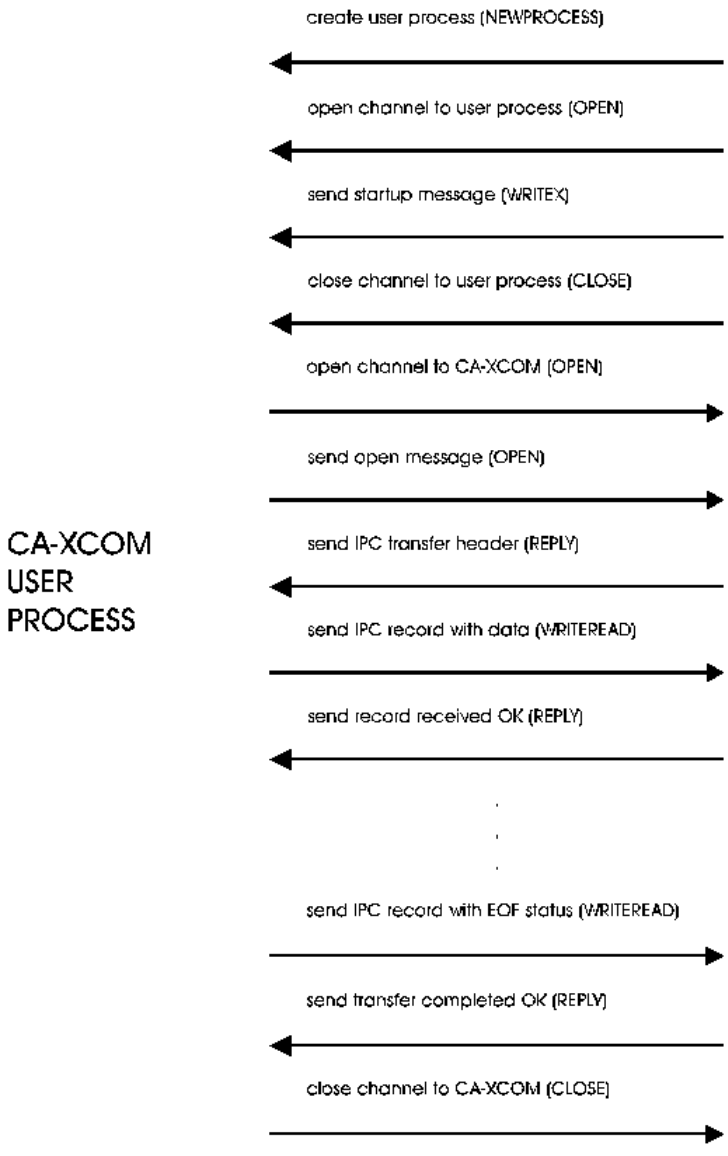
```
struct IPC_record_msg{
char   msgcode[2];                  /* always "RM" */
char   user_process[6];
char   CA-XCOM_process[6];
int status;
int data_size;
char data[data_size];
} IPC_record;
```

Data Flows

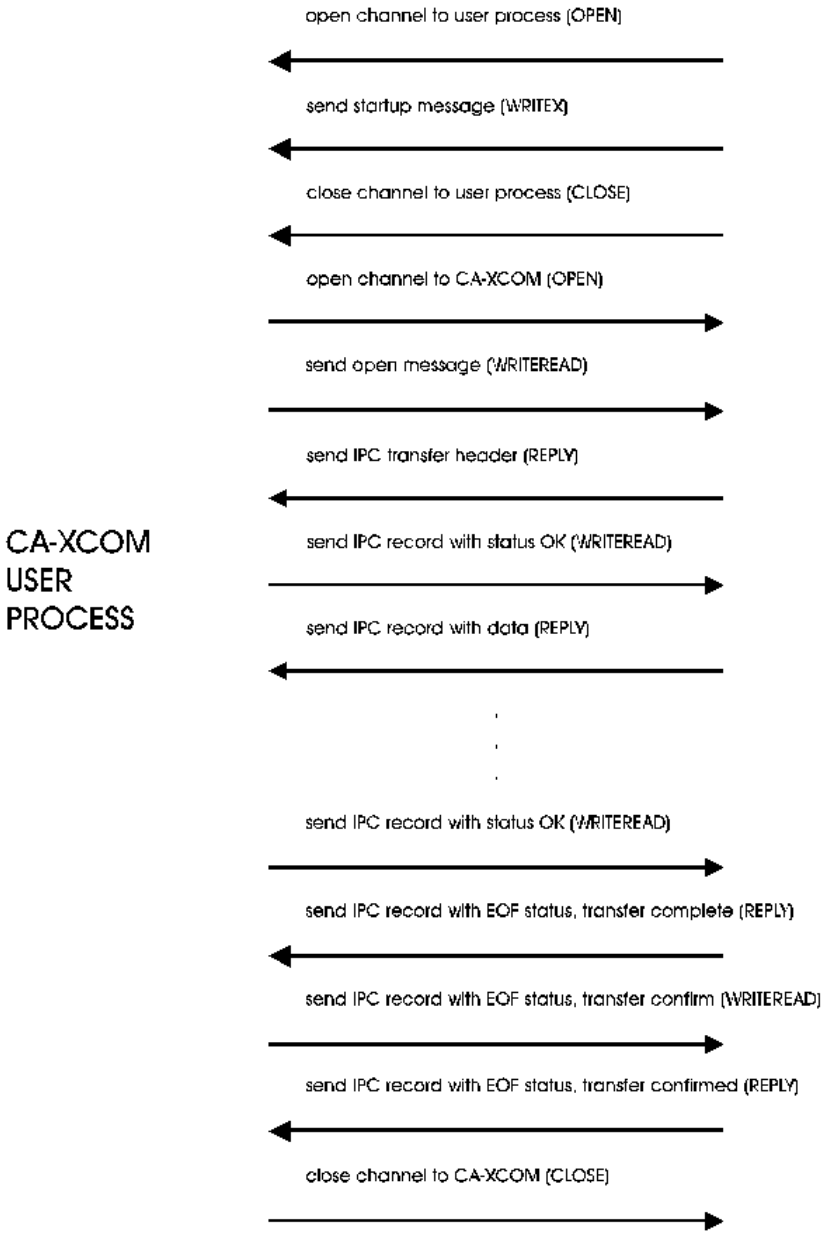
This section contains examples of data flows for the following IPC transactions:

- A command-line-initiated IPC send when the user process does not exist.
- A command-line-initiated IPC receive when a user process exists.
- An API-initiated IPC receive when a user process exists.

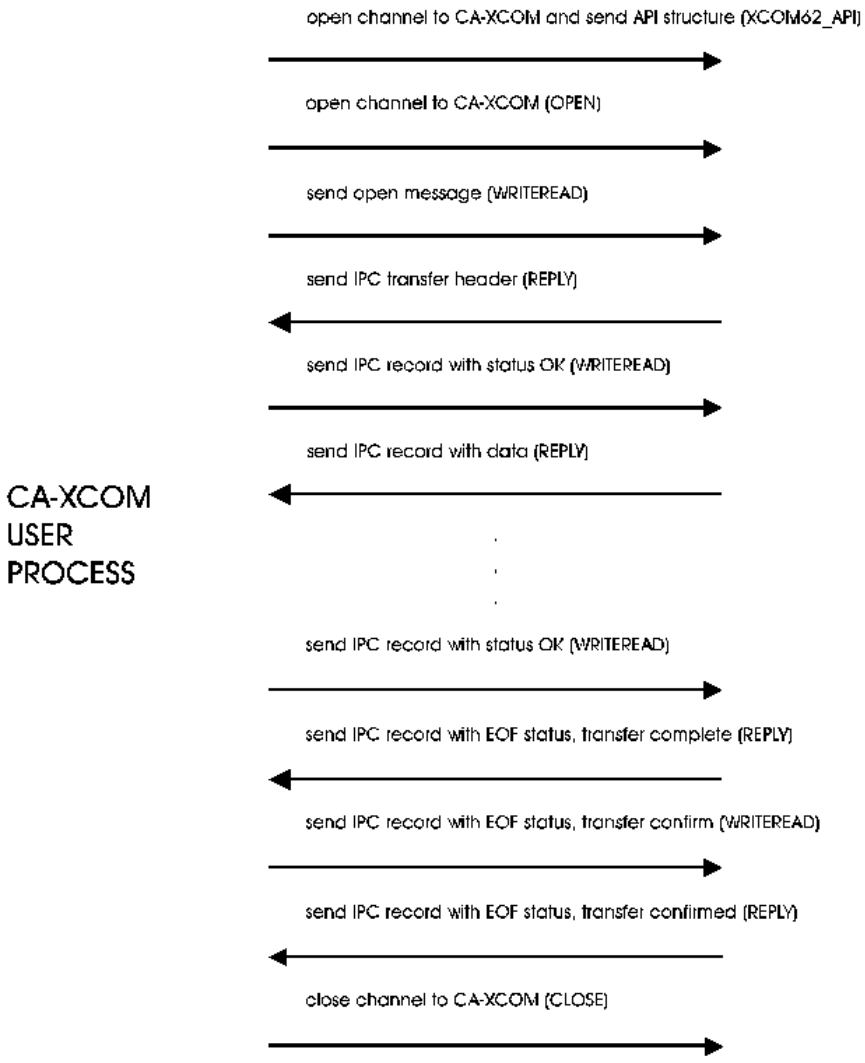
Command-Line-Initiated IPC Send When the User Process Does Not Exist



Command-Line-Initiated IPC Receive When a User Process Exists



API-Initiated IPC Receive When a User Process Exists



Logic Flows

This section contains examples of logic formats for the following transactions:

- An IPC send from the command line
- An IPC receive from the command line
- An API-initiated send with CA XCOM Data Transport parameters

Entering an IPC Send from the Command Line

The following example shows the format for specifying an IPC send from the CA XCOM Data Transport for HP NonStop command line:

```
XCOM62 put myfile as mvs.dir.file, mvscnf, blksize=8192,
IPC_PNAME = $MYAPP,& IPC_FNAME=$CLX01.EXAMPLE.MYAPPL PARM1, PARM2, PARM3
```

IPC Send Example: Standard CA XCOM Data Transport for HP NonStop Initialization

```
Validate IPC_PNAME
               not OK,  send error 494
Check for existence of $MYAPP
               not OK,  check if anything in IPC_FNAME
                              not OK,    send error 493
                              validate IPC_FNAME
                              not OK, send error 495

                              (NEWPROCESS)    run $CLX01.EXAMPLE.MYAPPL
                              not OK, send error 492

OK,     $MYAPP now exists
Open channel to $MYAPP  (OPEN)

Send startup message identifying CA-XCOM, if CA-XCOM started $MYAPP, also pass startup parameters from
IPC_FNAME parameter, (PARM1, PARM2, PARM3)   (WRITEX)

Close channel   (CLOSE)

Wait for IPC to open CA-XCOM's $RECEIVE          (READUPDATE on $RECEIVE)

CA-XCOM receives open message
            not OK,  send error 491              (user process WRITEREAD)
CA-XCOM replies with transfer header message     (REPLY)

CA-XCOM receives IPC record         (READUPDATE on $RECEIVE to user process WRITEREAD)

Check IPC record status
             not OK,  if status = 1, EOF, wait for transfer to complete and reply status
             not OK,  if status != 1, abort transfer, reply back with same status

Previous IPC record transmitted
             not OK,  reply CA-XCOM error code

Validate current IPC data
             not OK, reply error 497
Process data

Read next IPC record                (READUPDATE on $RECEIVE)
```

Entering an IPC Receive from the Command Line

The following example shows the format for specifying an IPC receive:

```
CA-XCOM62 get myfile as mvs.dir.file, mvscnf, blksize=8192, IPC_PNAME = $MYAPP,&
IPC_FNAME=$CLX01.EXAMPLE.MYAPPL PARM1, PARM2, PARM3
```

IPC Receive Example: Standard CA XCOM Data Transport for HP NonStop Initialization

```
Validate IPC_PNAME
              not OK,  send error 494

Check for existence of $MYAPP
              not OK,    check if  anything in IPC_FNAME
                                   not OK, send error 493
                                   validate IPC_FNAME
                                   not OK, send error 495

                                   (NEWPROCESS)      run $CLX01.EXAMPLE.MYAPPL
                                   not OK, send error 492

OK,  $MYAPP now exists
Open channel to $MYAPP   (OPEN)

Send startup message identifying CA-XCOM, if CA-XCOM started $MYAPP, also pass startup parameters from
IPC_FNAME parameter, (PARM1, PARM2, PARM3)  (WRITEX)

Close channel     (CLOSE)

Wait for IPC to open CA-XCOM's $RECEIVE         (READUPDATE on $RECEIVE)

CA-XCOM receives open message
           not OK,  send error 491              (user process WRITEREAD)
CA-XCOM replies with transfer header message    (REPLY)

CA-XCOM gets data from remote system

CA-XCOM receives IPC record         (READUPDATE on $RECEIVE to user process WRITEREAD)

Check IPC record status
            not OK,   abort transfer, reply back with same status

CA-XCOM has data to send
            not OK,  send IPC record with CA-XCOM error

CA-XCOM sends IPC record with data              (REPLY)

CA-XCOM gets data from remote system
```

Sending an API with CA XCOM Data Transport for HP NonStop Parameters

The example below shows how to open a channel to CA XCOM Data Transport for HP NonStop and send an API structure (CA-XCOM62_API) with the following CA XCOM Data Transport for HP NonStop parameters:

```
IPC_PNAME = $MYAPP

IPC_FNAME = $CLX01.EXAMPLE.MYAPPL PARM1, PARM2, PARM3
```

API Send Example:
Standard CA XCOM Data Transport for HP NonStop Initialization

```
Validate IPC_PNAME
               not OK,  send error 494

Check for existence of $MYAPP
               not OK,    continue processing transfer as in above examples

Wait for process $MYAPP to open CA-XCOM

continue processing transfer as in above examples
```

New Error Messages

501

IPC Start up message parameters don't match

502

Error starting IPC program, Newprocess error #nnn

503

IPC process does not exist, no IPC filename provided

504

Illegal IPC process name

505

Illegal IPC filename

506

IPC process Guardian I/O error #nnn

507

Data record length exceeds IPC header record/block size

API Structures for CA XCOM Data Transport for HP NonStop

Conversion Considerations

The API for this release of CA XCOM Data Transport for HP NonStop is quite different from the API of previous releases. To use your old application with this release, you must recompile with the new API structures and bind your program with the new APIO file.

Note: Your program must set the APIVERSION field to 3, or CA XCOM Data Transport for HP NonStop will reject your transfer with an error 520.

If your program does not set the VERSION value, CA XCOM Data Transport for HP NonStop uses the version number from the configuration file specified in the CONFIGFILE field. If CA XCOM Data Transport for HP NonStop cannot find this value, VERSION defaults to 2. If CA XCOM Data Transport for HP NonStop has a problem loading your program's API parameters, it returns an appropriate error to your program.

The new API C and TAL structures appear on the following pages.

API C Transfer Structure

The following is the CA XCOM Data Transport for HP NonStop API C structure:

```
#pragma section api_transfer
/*****************************************************************************/
/*                                                                           */
/*    This file was generated from apiddl                                    */
/*                                                                           */
/*****************************************************************************/
typedef struct
{
   short                          apiversion;
   char                           configfile[36];
   short                          wait;
   char                           appcopenname[16];
   char                           appcprocessname[16];
   char                           appctype;
   short                          command;
   char                           idest[128];
   char                           luname[18];
   char                           max_snax_iosize[6];
   char                           rluname[128];
   short                          version;
   char                           xdir[25];
   char                           xlogfile[36];
   char                           xmode[9];
   char                           ascebc[27];
   char                           bulkio[2];
   char                           cache_buf;
   char                           checkpoint_count[5];
   char                           checkpoint_file[27];
   char                           compression;
   char                           ebcasc[27];
   char                           fileaction;
   char                           history_file[27];
   char                           ipc_pname[25];
   char                           ipc_fname[151];
   char                           nullfill;
   char                           pack;
   char                           request_no[7];
   char                           rfile[255];
   char                           restart_flag;
   char                           sio;
   char                           transfer_id[11];
   char                           xbuffsize[7];
   char                           xfile[255];
   char                           carriageflag;
   char                           codeflag;
   char                           guardianfiletype[2];
   char                           prialloc[6];
   char                           recfm[4];
   char                           secalloc[6];
```

```
   char                          alloc_unit;
   char                          blksize[6];
   char                          lrecl[6];
   char                          system_user_data[11];
   char                          transfer_user_data[11];
   char                          volume[11];
   char                          xunit[11];
   char                          carriagecontrol;
   char                          chars[5];
   char                          copies[4];
   char                          disposition;
   char                          eol_classes[128];
   char                          fcb[5];
   char                          form[11];
   char                          holdflag;
   char                          reporttitle[22];
   char                          spoolflag;
   char                          xclass;
   char                          xdestination[22];
   char                          jobname[9];
   char                          jobnumber[9];
   char                          uic[10];
   char                          xwhen[15];
   char                          localnotify[65];
   char                          notify;
   char                          tso_notify[9];
   char                          who[13];
   char                          spoolcollector[25];
   char                          conv_security;
   char                          password[32];
   char                          password_file[27];
   char                          remoteuser[13];

   char                          start_date[9];
   char                          start_time[9];
   char                          retry_time[6];
   char                          retries[6];

/* volke01 xcomr11 new header parms start */
   char                          storcls[9];
   char                          datacls[9];
   char                          mgtclas[9];
   char                          dsntype[9];
   char                          expdate_flag;      /* future */
   char                          tape_label[4];
   char                          tape;
   char                          hfs_flag;          /* future */
   char                          xcomfullssl[4];
   char                          xcomshowcipher;    /* future */
```

```
   char                          den;
   char                          expdt[6];
   char                          retpd[5];
   char                          unitct[3];
   char                          volct[4];
   char                          volsq[4];
   char                          labelnum[5];
   char                          tapedisp;
   char                          codetabl[4];
   char                          seclabel[9];
   char                          lclntfyl;
   char                          rmtntfyl;
   char                          configssl[255];
   char                          trusted;
   char                          domain[16];
   char                          gatewayguid[37];
   char                          createdelete;
   char                          compress_pds;
   char                          ipprocessname[16];
   char                          port[6];
} api_transfer_def;
#pragma section api_commands
/*****************************************************************************/
/*                                                                           */
/*    This file was generated from apiddl                                    */
/*                                                                           */
/*****************************************************************************/
#define xcom_send_file 1
#define xcom_send_report 2
#define xcom_send_job 3
#define xcom_receive_file 4
#pragma section api_error_codes
/*****************************************************************************/
/*                                                                           */
/*    This file was generated from apiddl                                    */
/*                                                                           */
/*****************************************************************************/
/*   Error codes returned by XCOM62^API                                      */
#define error_allocating_send_buffer 283
#define error_forking 284
#define error_creating_pipe 285
#define error_setting_local_user_id 286
#define error_setting_remote_user_id 287
#define error_system_failed 288
#define error_command_failed 289
#define error_receiving_overlay 290
#define error_sending_overlay 291
#define error_sending_error 292
#define error_expecting_send_state 293
```

```
#define error_expecting_receive_state 294
#define error_command_line 295
#define error_deallocating 296
#define error_requesting_headr_confirm 297
#define error_allocate 298
#define error_local_attach 299
#define error_remote_attach 300
#define error_starting_appc 301
#define error_opening_input_file 302
#define error_sending_header 303
#define error_sending_maxlrecl 304
#define error_receiving_header 305
#define error_invalid_header 306
#define error_invalid_pack_option 307
#define error_reading_input_file 309
#define error_receive_error 310
#define error_sending_data 311
#define error_confirming_data 312
#define error_negative_data_confirm 313
#define error_sending_trailer 314
#define error_negative_trailer_confirm 315
#define error_received_from_remote 316
#define error_receive_fmh_7 317
#define error_reversing_line 318
#define error_confirming_checkpoint 319
#define error_confirmed_checkpoint 320
#define error_opening_checkpoint_file 321
#define error_writing_checkpoint_file 322
#define error_recv_receiving_header 401
#define error_header_invalid 402
#define error_opening_output_file 403
#define error_confirming_header 404
#define error_receiving_maxlrecl 405
#define error_receiving_feature_record 406
#define error_maxlrecl_invalid 407
#define error_feature_record_protocol 408
#define error_requesting_feature_cnfrm 409
#define error_sending_feature_record 410
#define error_confirm_feature_record 411
#define error_receiving_data 412
#define error_trailer_invalid 413
#define error_closing_output_file 414
#define error_receiving_trailer 415
#define error_writing_output_file 416
#define error_confirming_trailer 417
#define error_interrupt_received 418
#define error_user_not_found 419
#define error_user_id_out_of_range 420
#define error_group_id_out_of_range 421
```

```
#define error_login_incorrect 422
#define error_receive_file_descriptor 423
#define error_sending_file_descriptor 424
#define error_repositioning_file 425
#define error_restart_cnts_dont_match 426
#define error_starting_tp 427
#define error_no_sessions_available 435
#define error_tp_abended 436
#define error_input_buffer_too_small 437
#define error_truncation_not_allowed 438
#define error_maxlrecl_too_big 440
#define error_reqnos_dont_match 441
#define error_filenames_dont_match 442
#define error_groupnames_dont_match 443
#define error_invalid_packing_type 448
#define error_in_compression_type 449
#define error_memory_allocation 468
#define error_file_already_exists 470
#define error_ipc_params_dont_match 501
#define error_ipc_program_not_started 502
#define error_ipc_process_not_exist 503
#define error_ipc_process_name_illegal 504
#define error_ipc_program_name_illegal 505
#define error_ipc_process_error 506
#define error_ipc_data_rec_lg_too_big 507
#define error_missing_parameter 519
#define error_api_incorrect_version 520
```

API TAL Transfer Structure

The following is the CA XCOM Data Transport for HP NonStop API TAL structure:

```
?Section API^TRANSFER
?PAGE
!*****************************************************************************
!
!    This file was generated from apiddl
!
!*****************************************************************************
STRUCT       API^TRANSFER^DEF (*);
   BEGIN
   INT          APIVERSION;
   STRUCT       CONFIGFILE;
      BEGIN STRING BYTE [0:35]; END;
   INT          WAIT;
   STRUCT       APPCOPENNAME;
      BEGIN STRING BYTE [0:15]; END;
   STRUCT       APPCPROCESSNAME;
      BEGIN STRING BYTE [0:15]; END;
   STRING       APPCTYPE;
   FILLER       1;
   INT          COMMAND;
   STRUCT       IDEST;
      BEGIN STRING BYTE [0:127]; END;
   STRUCT       LUNAME;
      BEGIN STRING BYTE [0:17]; END;
   STRUCT       MAX^SNAX^IOSIZE;
      BEGIN STRING BYTE [0:5]; END;
   STRUCT       RLUNAME;
      BEGIN STRING BYTE [0:127]; END;
   INT          VERSION;
   STRUCT       XDIR;
      BEGIN STRING BYTE [0:24]; END;
   STRUCT       XLOGFILE;
      BEGIN STRING BYTE [0:35]; END;
   STRUCT       XMODE;
      BEGIN STRING BYTE [0:8]; END;
   STRUCT       ASCEBC;
      BEGIN STRING BYTE [0:26]; END;
   STRUCT       BULKIO;
      BEGIN STRING BYTE [0:1]; END;
   STRING       CACHE^BUF;
   STRUCT       CHECKPOINT^COUNT;
      BEGIN STRING BYTE [0:4]; END;
   STRUCT       CHECKPOINT^FILE;
      BEGIN STRING BYTE [0:26]; END;
   STRING       COMPRESSION;
   STRUCT       EBCASC;
      BEGIN STRING BYTE [0:26]; END;
   STRING       FILEACTION;
   STRUCT       HISTORY^FILE;
```

```
   BEGIN STRING BYTE [0:26]; END;
STRUCT       IPC^PNAME;
   BEGIN STRING BYTE [0:24]; END;
STRUCT       IPC^FNAME;
   BEGIN STRING BYTE [0:150]; END;
STRING       NULLFILL;
STRING       PACK;
STRUCT       REQUEST^NO;
   BEGIN STRING BYTE [0:6]; END;
STRUCT       RFILE;
   BEGIN STRING BYTE [0:254]; END;
STRING       RESTART^FLAG;
STRING       SIO;
STRUCT       TRANSFER^ID;
   BEGIN STRING BYTE [0:10]; END;
STRUCT       XBUFFSIZE;
   BEGIN STRING BYTE [0:6]; END;
STRUCT       XFILE;
   BEGIN STRING BYTE [0:254]; END;
STRING       CARRIAGEFLAG;
STRING       CODEFLAG;
STRUCT       GUARDIANFILETYPE;
   BEGIN STRING BYTE [0:1]; END;
STRUCT       PRIALLOC;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       RECFM;
   BEGIN STRING BYTE [0:3]; END;
STRUCT       SECALLOC;
   BEGIN STRING BYTE [0:5]; END;
STRING       ALLOC^UNIT;
STRUCT       BLKSIZE;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       LRECL;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       SYSTEM^USER^DATA;
   BEGIN STRING BYTE [0:10]; END;
STRUCT       TRANSFER^USER^DATA;
   BEGIN STRING BYTE [0:10]; END;
STRUCT       VOLUME;
   BEGIN STRING BYTE [0:10]; END;
STRUCT       XUNIT;
   BEGIN STRING BYTE [0:10]; END;
STRING       CARRIAGECONTROL;
STRUCT       CHARS;
   BEGIN STRING BYTE [0:4]; END;
STRUCT       COPIES;
   BEGIN STRING BYTE [0:3]; END;
STRING       DISPOSITION;
STRUCT       EOL^CLASSES;
```

```
   BEGIN STRING BYTE [0:127]; END;
STRUCT       FCB;
   BEGIN STRING BYTE [0:4]; END;
STRUCT       FORM;
   BEGIN STRING BYTE [0:10]; END;
STRING       HOLDFLAG;
STRUCT       REPORTTITLE;
   BEGIN STRING BYTE [0:21]; END;
STRING       SPOOLFLAG;
STRING       XCLASS;
STRUCT       XDESTINATION;
   BEGIN STRING BYTE [0:21]; END;
STRUCT       JOBNAME;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       JOBNUMBER;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       UIC;
   BEGIN STRING BYTE [0:9]; END;
STRUCT       XWHEN;
   BEGIN STRING BYTE [0:14]; END;
STRUCT       LOCALNOTIFY;
   BEGIN STRING BYTE [0:64]; END;
STRING       NOTIFY;
STRUCT       TSO^NOTIFY;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       WHO;
   BEGIN STRING BYTE [0:12]; END;
STRUCT       SPOOLCOLLECTOR;
   BEGIN STRING BYTE [0:24]; END;
STRING       CONV^SECURITY;
STRUCT       PASSWORD;
   BEGIN STRING BYTE [0:31]; END;
STRUCT       PASSWORD^FILE;
   BEGIN STRING BYTE [0:26]; END;
STRUCT       REMOTEUSER;
   BEGIN STRING BYTE [0:12]; END;

STRUCT       START^DATE;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       START^TIME;
   BEGIN STRING BYTE [0:8]; END;
STRUCT       RETRY^TIME;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       RETRIES;
   BEGIN STRING BYTE [0:5]; END;
STRUCT       STORCLS;
    BEGIN STRING BYTE [0:8]; END;
STRUCT       DATACLS;
    BEGIN STRING BYTE [0:8]; END;
```

```
   STRUCT       MGTCLAS;
       BEGIN STRING BYTE [0:8]; END;
   STRUCT       DSNTYPE;
       BEGIN STRING BYTE [0:8]; END;
   STRING       EXPDATE^FLAG;         !* future
   STRUCT       TAPE^LABEL;
       BEGIN STRING BYTE [0:3]; END;
   STRING       TAPE;
   STRING       HFS^FLAG;             !* future
   STRUCT       XCOMFULLSSL;
       BEGIN STRING BYTE [0:3]; END;
   STRING       XCOMSHOWCIPHER;       !* future
   STRING       DEN;
   STRUCT       EXPDT;
       BEGIN STRING BYTE [0:5]; END;
   STRUCT       RETPD;
       BEGIN STRING BYTE [0:4]; END;
   STRUCT       UNITCT;
       BEGIN STRING BYTE [0:2]; END;
   STRUCT       VOLCT;
       BEGIN STRING BYTE [0:3]; END;
   STRUCT       VOLSQ;
       BEGIN STRING BYTE [0:3]; END;
   STRUCT       LABELNUM;
       BEGIN STRING BYTE [0:4]; END;
   STRING       TAPEDISP;
   STRUCT       CODETABL;
       BEGIN STRING BYTE [0:3]; END;
   STRUCT       SECLABEL;
       BEGIN STRING BYTE [0:8]; END;
   STRING       LCLNTFYL;
   STRING       RMTNTFYL;
   STRUCT       CONFIGSSL;
       BEGIN STRING BYTE [0:254]; END;
   STRING       TRUSTED;
   STRUCT       DOMAIN;
        BEGIN STRING BYTE [0:15]; END;
   STRUCT       GATEWAYGUID;
         BEGIN STRING BYTE [0:36]; END;
   STRING       CREATEDELETE;
   STRING       COMPRESS^PDS;
   STRUCT       IPPROCESSNAME;
       BEGIN STRING BYTE [0:5]; END;
   STRUCT       PORT;
       BEGIN STRING BYTE [0:5]; END;
   END;
?Section API^COMMANDS
!*****************************************************************************
!
```

```
!    This file was generated from apiddl
!
!*****************************************************************************
Literal XCOM^SEND^FILE = 1;
Literal XCOM^SEND^REPORT = 2;
Literal XCOM^SEND^JOB = 3;
Literal XCOM^RECEIVE^FILE = 4;
?Section API^ERROR^CODES
!*****************************************************************************
!
!    This file was generated from apiddl
!
!*****************************************************************************
!   Error codes returned by XCOM62^API
Literal ERROR^ALLOCATING^SEND^BUFFER = 283;
Literal ERROR^FORKING = 284;
Literal ERROR^CREATING^PIPE = 285;
Literal ERROR^SETTING^LOCAL^USER^ID = 286;
Literal ERROR^SETTING^REMOTE^USER^ID = 287;
Literal ERROR^SYSTEM^FAILED = 288;
Literal ERROR^COMMAND^FAILED = 289;
Literal ERROR^RECEIVING^OVERLAY = 290;
Literal ERROR^SENDING^OVERLAY = 291;
Literal ERROR^SENDING^ERROR = 292;
Literal ERROR^EXPECTING^SEND^STATE = 293;
Literal ERROR^EXPECTING^RECEIVE^STATE = 294;
Literal ERROR^COMMAND^LINE = 295;
Literal ERROR^DEALLOCATING = 296;
Literal ERROR^REQUESTING^HEADR^CONFIRM = 297;
Literal ERROR^ALLOCATE = 298;
Literal ERROR^LOCAL^ATTACH = 299;
Literal ERROR^REMOTE^ATTACH = 300;
Literal ERROR^STARTING^APPC = 301;
Literal ERROR^OPENING^INPUT^FILE = 302;
Literal ERROR^SENDING^HEADER = 303;
Literal ERROR^SENDING^MAXLRECL = 304;
Literal ERROR^RECEIVING^HEADER = 305;
Literal ERROR^INVALID^HEADER = 306;
Literal ERROR^INVALID^PACK^OPTION = 307;
Literal ERROR^READING^INPUT^FILE = 309;
Literal ERROR^RECEIVE^ERROR = 310;
Literal ERROR^SENDING^DATA = 311;
Literal ERROR^CONFIRMING^DATA = 312;
Literal ERROR^NEGATIVE^DATA^CONFIRM = 313;
Literal ERROR^SENDING^TRAILER = 314;
Literal ERROR^NEGATIVE^TRAILER^CONFIRM = 315;
Literal ERROR^RECEIVED^FROM^REMOTE = 316;
Literal ERROR^RECEIVE^FMH^7 = 317;
Literal ERROR^REVERSING^LINE = 318;
```

```
Literal ERROR^CONFIRMING^CHECKPOINT = 319;
Literal ERROR^CONFIRMED^CHECKPOINT = 320;
Literal ERROR^OPENING^CHECKPOINT^FILE = 321;
Literal ERROR^WRITING^CHECKPOINT^FILE = 322;
Literal ERROR^RECV^RECEIVING^HEADER = 401;
Literal ERROR^HEADER^INVALID = 402;
Literal ERROR^OPENING^OUTPUT^FILE = 403;
Literal ERROR^CONFIRMING^HEADER = 404;
Literal ERROR^RECEIVING^MAXLRECL = 405;
Literal ERROR^RECEIVING^FEATURE^RECORD = 406;
Literal ERROR^MAXLRECL^INVALID = 407;
Literal ERROR^FEATURE^RECORD^PROTOCOL = 408;
Literal ERROR^REQUESTING^FEATURE^CNFRM = 409;
Literal ERROR^SENDING^FEATURE^RECORD = 410;
Literal ERROR^CONFIRM^FEATURE^RECORD = 411;
Literal ERROR^RECEIVING^DATA = 412;
Literal ERROR^TRAILER^INVALID = 413;
Literal ERROR^CLOSING^OUTPUT^FILE = 414;
Literal ERROR^RECEIVING^TRAILER = 415;
Literal ERROR^WRITING^OUTPUT^FILE = 416;
Literal ERROR^CONFIRMING^TRAILER = 417;
Literal ERROR^INTERRUPT^RECEIVED = 418;
Literal ERROR^USER^NOT^FOUND = 419;
Literal ERROR^USER^ID^OUT^OF^RANGE = 420;
Literal ERROR^GROUP^ID^OUT^OF^RANGE = 421;
Literal ERROR^LOGIN^INCORRECT = 422;
Literal ERROR^RECEIVE^FILE^DESCRIPTOR = 423;
Literal ERROR^SENDING^FILE^DESCRIPTOR = 424;
Literal ERROR^REPOSITIONING^FILE = 425;
Literal ERROR^RESTART^CNTS^DONT^MATCH = 426;
Literal ERROR^STARTING^TP = 427;
Literal ERROR^NO^SESSIONS^AVAILABLE = 435;
Literal ERROR^TP^ABENDED = 436;
Literal ERROR^INPUT^BUFFER^TOO^SMALL = 437;
Literal ERROR^TRUNCATION^NOT^ALLOWED = 438;
Literal ERROR^MAXLRECL^TOO^BIG = 440;
Literal ERROR^REQNOS^DONT^MATCH = 441;
Literal ERROR^FILENAMES^DONT^MATCH = 442;
Literal ERROR^GROUPNAMES^DONT^MATCH = 443;
Literal ERROR^INVALID^PACKING^TYPE = 448;
Literal ERROR^IN^COMPRESSION^TYPE = 449;
Literal ERROR^MEMORY^ALLOCATION = 468;
Literal ERROR^FILE^ALREADY^EXISTS = 470;
Literal ERROR^IPC^PARAMS^DONT^MATCH = 501;
Literal ERROR^IPC^PROGRAM^NOT^STARTED = 502;
Literal ERROR^IPC^PROCESS^NOT^EXIST = 503;
Literal ERROR^IPC^PROCESS^NAME^ILLEGAL = 504;
Literal ERROR^IPC^PROGRAM^NAME^ILLEGAL = 505;
Literal ERROR^IPC^PROCESS^ERROR = 506;
```

```
Literal ERROR^IPC^DATA^REC^LG^TOO^BIG = 507;
Literal ERROR^MISSING^PARAMETER = 519;
Literal ERROR^API^INCORRECT^VERSION = 520;
```

Chapter 8: Remote Spooling

Xque provides both printer sharing and report distribution among similar and dissimilar systems. By configuring a CA XCOM Data Transport print process for each remote printer, CA XCOM Data Transport for HP NonStop can remotely print files from HP NonStop using the standard HP NonStop spooler facilities.

Xque can also send the HP NonStop spool jobs as files to similar and dissimilar computer file systems.

As a result, you can:

- Print or send spool files as files remotely using any HP NonStop command that sends output to a spooler.
- Review files on the spooler using the SPOOLCOM and PERUSE programs.
- Have applications use Xque to write remote reports directly to remote printers or file systems.

Note: Before attempting to use CA XCOM Data Transport for HP NonStop, ensure that the SNAX/APC and/or TCP/IP processes are started.

This section contains the following topics:

The Xque Process

Xque sends reports to remote systems using standard operating system commands to put files on standard operating system print queues or file systems. This process consists of the following four stages:

- The CA XCOM Data Transport for HP NonStop provided spooler program XCOMPRNT receives requests from the HP NonStop spooler.
- XCOMPRNT creates an XCOM62 process to do the actual transfer to the remote system.
- XCOMPRNT reads the spooled file and passes each record to the XCOM62 process.
- XCOM62 transfers the file to the remote system.

XCOMPRNT is an HP NonStop spooler print process provided to automatically transfer reports from the HP NonStop spool to a remote system as a report for printing or as a file. This provides the same functionality as Process SYSOUT in the CA XCOM Data Transport for Z/OS product.

The Sample Spooler Startup Files

Xque is defined using spooler startup files in which the print process, the location, and the device are defined. The following files in the CA XCOM Data Transport for HP NonStop release subvolume are sample spooler startup files:

- ZSPLCOLD
- ZSPLCONF
- ZSPLWARM

Configure Your System for Remote Printing

The following sections describe the master procedure and detailed instructions for each step in configuring your system for remote printing.

The Master Procedure

To configure your system for remote printing

1. Add DEFINE statements to the environment.
2. Configure the spooler cold start file.
3. Create a configuration file for each remote system.

Step 1: Add DEFINE Statements to the Environment

Because XCOMPRNT starts up CA XCOM Data Transport to transfer a file from the spooler to a remote printer, you must define the location of CA XCOM Data Transport in the environment.

XCOMPRNT uses this define to locate the XCOM62 program when it starts the CA XCOM Data Transport process:

```
ADD DEFINE =XCOM62-PROGRAM,CLASS MAP,FILE volume.subvolume.XCOM62
```

The variable volume.subvolume is the location of CA XCOM Data Transport for HP NonStop.

You must also add defines (listed below) to the environment. These defines enable Tandem's Event Management Service routines to monitor program events and to identify the event collector where the EMS messages are written.

```
ADD DEFINE =EGEN_ADD_EVENT_TEXT,CLASS MAP,FILE $YES
ADD DEFINE =_EMS_COLLECTOR,CLASS MAP,FILE eventcollector
```

The variable *eventcollector* specifies the name of the event collector that you want to write to. If you do not specify this define, the default system primary collector ($0) is used.

The sample spooler startup file ZSPLCOLD starts a process called $SCIS. $SCIS invokes SPOOLCOM to execute the CA XCOM Data Transport for HP NonStop spooler configuration file ZSPLCONF. A printed copy of the sample ZSPLCOLD file is as follows:

```
#FRAME
COMMENT ***  STOP THE SUPERVISOR PROCESS ***
[#IF [#processexists $scis] |THEN|
    SPOOLCOM $scis; SPOOLER, DRAIN
    DELAY 15 seconds
]
COMMENT ***  PURGE EXISTING DATA AND CONTROL FILES ***
fup purge $clx12.scispl.* !
COMMENT ***  CREATE THE SPOOLER COLLECTOR DATA FILES ***
fup create $dsmscm.xctndm11.data,ext (128,128)
COMMENT ***  Add defines to specify the location of the XCOM62 program
COMMENT ***    the EMS collector process
COMMENT ***    whether to include text in ems messages
SINK [#DEFINEDELETE =XCOM62-PROGRAM]
add define =XCOM62-PROGRAM, class map, file $dsmscm.xctndm11.xcom62
SINK [#DEFINEDELETE =_EMS_COLLECTOR]
add define =_EMS_COLLECTOR, class map, file $0
SINK [#DEFINEDELETE =_EGEN_ADD_EVENT_TEXT]
add define =_EGEN_ADD_EVENT_TEXT, class map, file $YES
COMMENT *** RUN THE SUPERVISOR ***
spool/pri 149,name $scis,nowait,in $clx12.scispl.spl,out $0,cpu 1&
```

```
/2,1500,500,10,4,10
spool/pri 149,name $scis,nowait,in $clx12.scispl.spl,out $0,cpu 1&
/3,1500,500,10,4,10
COMMENT --  control file                  = $xctndm11.SPL(0-4)
COMMENT --  log file                      = $0 (OPERATING CONSOLE)
COMMENT --  supervisor process name       = $SCIS
COMMENT --  execution priority            = 149
COMMENT --  maximum number of jobs        = 1500
COMMENT --  maximum number of locations   = 500
COMMENT --  maximum number of devices     = 10
COMMENT --  maximum number of collectors  = 4
COMMENT --  maximum number of print procs = 10
COMMENT *** THE SPOOLER IS NOW IN A COLD STATE.  THE NEXT COMMAND RUNS ***
COMMENT *** SPOOLCOM, SPECIFYING THE SPOOLER CONFIGURATION FILE TO ***
COMMENT *** INITIALIZE AND START THE SPOOLER. ***
run $system.system.spoolcom / in $dsmscm.xctndm11.zsplconf / $scis
#UNFRAME
```

Step 2: Configure the Spooler Cold Start File

After you add the DEFINE statements, a startup file called ZSPLCONF must be configured for Xque. In this startup file, you must add a print process, a device, and a location for each remote printer that you want to use.

Example:

The sample spooler file ZSPLCONF with these definitions included is as follows:

```
COMMENT  *** A spooler collector file named $J
COLLECT $J,FILE $system.system.cspool,PRI 152,CPU 1,BACKUP 2
COLLECT $J,DATA $xata3.scispl.data,UNIT 3
COMMENT  ***  Add the CA XCOM Data Transport Print Process
PRINT $scp1, cpu 2,debug off, file $xata3.sci.xcomprnt, parm 0, pri 145
PRINT $scp2, cpu 2,debug off, file $xata3.sci.xcomprnt, parm 0, pri 145
COMMENT  ***  Printer Devices
COMMENT      sci.stratcnf and sci.ibmvscnf are the names of CA XCOM Data Transport
configuration
COMMENT       files.
DEV $dsmscm.xctndm11.stratcnf,PROCESS $scp1,SPEED 100,FIFO,HEADER OFF,EXCLUSIVE ON,
RESTART 180, TIMEOUT -1
DEV $dsmscm.xctndm11.ibmvscnf,PROCESS $scp2,SPEED 100,FIFO,HEADER OFF,EXCLUSIVE
OFF!, RESTART 180, TIMEOUT -1
COMMENT  ***  Locations
LOC #stratus.default ,DEV $dsmscm.xctndm11.stratcnf
LOC #ibm.default     ,DEV $dsmscm.xctndm11.ibmvscnf
COMMENT  ***  Start the spooler
SPOOLER,START
```

Print Process

Before you start the spooler, you must add the print process. The following sample SPOOLCOM command adds a CA XCOM Data Transport for HP NonStop print process to the spooler:

```
PRINT $XCOM1, CPU2, FILE $SYSTEM.SYSTEM.XCOMPRNT, PRI 145
```

Printer Devices

The device names for remote printing are the names of the CA XCOM Data Transport for HP NonStop configuration files configured specifically for the spool devices that XCOMPRNT uses. Each file contains the parameters that describe how to transfer reports to the remote system.

The print process XCOMPRNT uses the DEVICE name defined to the HP NonStop spooler to define the CA XCOM Data Transport configuration file. This configuration file must contain the information required to send a file to the desired remote system, plus the printer-specific CA XCOM Data Transport Send Report parameters or the file specific Send File parameters.

Locations

You must define a location for your spooler that is associated with your pseudo CA XCOM Data Transport device. This location is the target for your local print commands. After you have defined this location, whenever you want to send a file to a remote system, you can specify a HP NonStop PRINT command to send a file to the location defined for that printer:

```
FUP COPY MYFILE.OUT, spooler.location
```

For example, your *spooler.location* might be $J.#IBM.DEFAULT.

Step 3: Create a Configuration File for Each Remote Printer

The spooler startup file invokes a configuration file for each system. These configuration files, which govern the remote spooling transfer, must contain the following parameter types:

- As a report:
  - Remote destination
  - Send report
  - File creation
  - General transfer
  - Notification and security
- As a file:
  - Remote destination
  - Send File
  - File creation
  - General transfer
  - Notification and security

Sample configuration spool device files are provided for the XQUE Remote Spooling feature, as follows:

- XQUERCNF is a sample for sending spool files as a REPORT on a remote system.
- XQUEFCNF is a sample for sending spool files as a FILE to a remote system.

Remote Destination Parameters

The following sections describe the remote destination parameters.

APPC_PROCESS_NAME

The name of the process used by CA XCOM Data Transport. This name must agree with the process name specified in the SNAX/APC configuration.

Example:

If you used the supplied PATHCOLD file to start SNAX/APC, the APPC_PROCESS_NAME is $SNAS.

Range: Up to 16 characters

Default: None

APPC_TYPE

Required.

Indicates your APPC configuration type.

Range: SNAXAPPC or TCPIP

Default: SNAXAPPC

IO_BUFFSIZE

Used with SNAX/APC.

Specifies the size of the buffer passed between CA XCOM Data Transport for HP NonStop and SNAX/APC. The IO_BUFFSIZE value must be less than or equal to the SNAX/APC MAXAPPLIDSIZE parameter value.

CA XCOM Data Transport for HP NonStop fills a buffer of this size before sending to SNAX/APC. When receiving files, SNAX/APC returns a buffer of this size to CA XCOM Data Transport.

IO_BUFFSIZE lets you maximize throughput and eliminate excessive overhead in interprocess communication between CA XCOM Data Transport for HP NonStop and SNAX/APC. In general, this parameter value is larger for higher speed lines.

Range: Values between 4136 and 32000

Default: 4136

PORT

For TCP/IP, this is the port number for CA XCOM Data Transport on the remote system.

REMOTE_SYSTEM

The name of the remote LU as configured in the APPC:

For SNAX

The remote SNA LU name.

For TCP/IP

The remote TCP/IP name or address.

Range: Up to 128 characters

Default: None

VERSION

Locally initiated transfers only.

Indicates whether the request is a Version 1 or Version 2 transfer.

1

Version 1

2

Version 2

Default: 2

XIDEST

This parameter is used to specify the LU name of the IBM mainframe to which the file is to be directly transferred. The mainframe LU name must be configured in SNAX/APC. If this variable is null or unset, a direct connection is attempted to the remote system.

Range: Up to 21 characters

Default: None

XDIR

Specifies the default volume and subvolume for all files read and written by CA XCOM Data Transport for HP NonStop except the XCOMCNF file.

Note: If a transfer is initiated remotely and this parameter is not specified, the default volume of the remote system's user ID is used.

Range: Up to 256 characters

Default: None

XLOGFILE

For locally initiated CA XCOM Data Transport transfers only.

Provides a file name for the log file for locally initiated transfers.

Range: Up to 250 characters.

Default: XCOMLOG

XLUNAME

Identifies the local LU name to use during this transmission.

Range: Up to eight characters

Default: None

XMODE

Specifies the mode name that CA XCOM Data Transport for HP NonStop will use during this transmission.

Range: Up to eight characters

Default: XCOMMODE

Send Report Parameters

The following sections describe the send report parameters.

CARRIAGE_CONTROL_CHARACTERS

Indicates which carriage control characters are used in the print job.

ASA

ASA control codes in column 1.

IBM

IBM Machine Characters (for z/OS only).

OTHER

Uses no carriage control codes.

Note: Set this parameter value to ASA to preserve page skips and spacing.

Default: OTHER

CHARS

Reports only.

Specifies the font for reports sent to a z/OS system. For more information, see your z/OS manual.

Range: Up to four characters

Default: None

CLASS

Reports only.

Indicates the print class for the print job.

If the remote system is a z/OS system, then CLASS designates the JES SYSOUT class. In this case, to print the report through SYSOUT=B, enter B.

Note: If printing on HP NonStop, this parameter is ignored.

Range: One character

Default: None

COPIES

Indicates the number of copies to be printed.

Range: 0 to 999

Default: 1

DESTINATION

Indicates the destination on the remote system for the print job. If no destination is specified, the job is sent to the system's default printer.

For report printing on HP NonStop, the remote system should specify the destination as #location-name. The spool collector name is specified in that parameter.

Range: Up to 16 characters

Default: None

DISPOSITION

Reports only.

Indicates the disposition of the printed file after printing is completed. Whether this parameter is meaningful is system dependent.

DELETE

Delete the file after it is printed.

KEEP

Do not delete the file.

HOLD

Hold after printing.

Default: DELETE

FCB

Indicates the forms control block (FCB) JCL parameter when sending the report file to a z/OS mainframe. It defines print density, lines per page, and so on.

Note: FCB is ignored for report printing on HP NonStop systems.

Range: Up to four characters

Default: None

FORM

Specifies which form the printed output should use.

Because CA XCOM Data Transport places the print job in the remote system's print queue, the print control functions depend on the remote system. Before sending a report, you must verify that the type of form you are requesting is available at the remote site.

Note: When sending a report to an OpenVMS system, leave the FORM parameter blank unless you are certain that you are entering a valid form type value. OpenVMS interprets a blank to mean that no special form is being requested.

Range: Up to 10 characters

Default: The remote printer's form

HOLD_FLAG

Indicates the transferred report file's printing status on the remote system.

YES

Place the file on HOLD on the remote system.

NO

Prepare the file for immediate printing.

Default: NO

SPOOL_FLAG

System-dependent flag.

Indicates to the remote system whether it should spool the report received. HP NonStop sends all the reports that it receives to the spooler.

YES

Spool the report received from the local system.

NO

Do not spool the report.

Default: YES

Send File Parameters

The following list sections describe the send report parameters.

ALLOC_UNIT

Used only when creating mainframe files.

Specifies the size of the allocation unit if the remote is an IBM mainframe. The actual byte count of each type will vary, depending on the storage device.

B

Blocks

C

Cylinders

T

Tracks

Default: B

Note: If you have questions about allocation units, consult your System Administrator.

BLKSIZE

Specifies the physical block size of a file. The range depends on record length.

For a variable record format

BLKSIZE = LRECL + 4

For a fixed or fixed blocked record format

BLKSIZE = a multiple of LRECL

For an undefined record format

BLKSIZE > largest record length

Note: If you create a structured file on the HP NonStop system, it must be a valid HP NonStop block size. CA XCOM Data Transport computes an appropriate value.

Range: Up to five characters

Default: 4096

DIR_ALLOC

Specifies the number of directory blocks to allocate when creating a PDS data set on a remote z/OS system. This corresponds to MAXEXTENTS on HP NonStop.

Range: 0 to 32767

Default: 0

FILE_OPTION

Indicates how the transferred data is to be processed by the receiving system.

CREATE

Creates a new file on the receiving system.

APPEND

Appends this data to an existing file on the receiving system.

REPLACE

Replaces the contents of an existing file on the receiving system. On HP NonStop, if the file does not exist, it is created automatically.

Default: CREATE

PRI_ALLOC

The primary extent size for creating local and remote files.

Range: Up to five characters

Default: 2

RECORD_FORMAT

Specifies the record format of a data set created on an IBM mainframe. This corresponds to the JCL RECFM subparameter.

F (Fixed Unblocked)

All records have the same length.

FB (Fixed Blocked)

Fixed record length with multiple records per block.

VB (Variable Blocked)

Variable record length with multiple records per block.

U (Undefined)

Undefined record length.

Default: VB

REMOTE_FILE

Indicates the file on the remote system to which the transferred data is being written. If you are creating the file (FILE_OPTION=CREATE), the file name must be consistent with the file naming conventions of the remote system.

The local CA XCOM Data Transport system does not validate this name. The remote I/O system determines whether the file name is valid.

Note: For send file transfers only

Range: Up to 256 characters

Default: None

SEC_ALLOC

The secondary extent size for creating local and remote files.

Range: 1 to 3567

Default: 4

UNIT

Intended for specifying to a remote system (primarily an IBM mainframe) the unit that a data set is to be created on.

Range: Up to six characters

Default: None

VOLUME

Specifies the volume on which a data set is to be created on an IBM mainframe.

Range: Up to six characters

Default: None

XQUE_FILE

Specifies a prefix that to be used, along with the HP NonStop spool files seven-character job number prefixed with a J, to build a unique REMOTE_FILE file name.

This process is overridden if a value for the REMOTE_FILE parameter is specified.

Note: For xque send file transfers only

Range: Up to 248 characters

Default: None

File Creation Parameters

The following sections describe the file creation parameters.

CARRIAGE_FLAG

Controls the treatment of text files

If CARRIAGE_FLAG=YES and CODE_FLAG is ASCII or EBCDIC, new line characters are added to incoming records and removed from outgoing records.

Range: YES or NO

Default: YES

CODE_FLAG

Identifies the type of data being transferred so that CA XCOM Data Transport for HP NonStop knows if it should translate the data.

B

A binary file such as an executable file is being transferred. No translation required.

A

An ASCII file is being transferred. No translation required.

E

If HP NonStop is sending the file, it translates it into EBCDIC; if HP NonStop is receiving the file, it translates it back to ASCII.

Default: EBCDIC

Note: CA XCOM Data Transport for HP NonStop translates every byte in the file. If you have mixed characters and binary data, the file will be corrupted if you specify EBCDIC.

LRECL

Specifies the actual or maximum length in bytes of a logical record. This corresponds to the JCL LRECL subparameter.

For a variable blocked format

LRECL should equal the maximum record length.

For a fixed or fixed blocked format

LRECL should equal the constant record length.

Range: Up to five characters

Default: 0, except in the following cases:

- If GUARDIAN_FILE_TYPE=EDIT or UNSTRUCTURED, the default is 239, or 243 for variable blocked.
- If GUARDIAN_FILE_TYPE=RELATIVE or ENTRYSEQ, the default is taken from the record length parameter in the transferred file.

Notification and Security Parameters

The following sections describe the notification and security parameters.

LOCAL_NOTIFY

Specifies which user to notify on the local system when CA XCOM Data Transport has completed the transfer.

Range: Up to 64 characters

Default: None

NOTIFYR

Specifies the notification flag on the remote system.

TSO

TSO user notification.

WTO

Write to log only.

CICS

CICS user notification.

LU

Logical unit notification.

VM

VM/CMS user notification.

NONE

No user notification.

Note: This parameter is associated with the NOTIFY_NAME parameter.

Default: LU

NOTIFY_NAME

Specifies which user to notify on the remote system when CA XCOM Data Transport has completed its procedure.

If the remote system is a z/OS system, CA XCOM Data Transport uses the value of NOTIFYR to determine the type of notification to deliver.

If the remote system is an HP NonStop system, the user receives a mail message.

Range: Up to 12 characters

Default: None

PASSWORD

Indicates the remote password to use with the file security scheme on the remote system.

Range: Up to 31 characters

Default: None

PASSWORD_FILE

Specifies the name of the CA XCOM Data Transport security file. Use the following format:

vol.subvol.filename

Range: Up to 27 characters in the specified format or NONE, which disables the security feature.

Default: NONE

USERID

Identifies the remote user ID for use with the file security scheme on the remote system.

Range: Up to 12 characters

Default: None

General Transfer Parameters

The following sections describe the notification and security parameters.

TRANSFER_USER_DATA

Indicates any user-specified information for each transfer that can be passed to the remote system. This information is written in the history record.

Range: Up to 10 characters

Default: None

SYSTEM_USER_DATA

Specifies system-dependent user data included with each transfer. Only used with Version 2 partners.

Range: Up to 10 characters

Default: None

XTRACE

Indicates the trace level.

Range: 0 to 9

0

No tracing.

9 (maximum)

Outputs the raw contents of data buffers.

Default: 0

Using Xque

CA XCOM Data Transport for HP NonStop can be used to print locally generated reports such as ENFORM reports and SQLCI output compiler listings on a remote system. After the spooler has been configured, output can be directed to a remote printer by specifying the location of the listing file.

Any Tandem program that generates output can send an Xque report. Xque reports can consist of the following:

- The output of a compile
- PTRACE output
- FUP-specified output
- ENFORM-specified output

Example:

The following ENFORM command causes the report DAILYRPT to print on a remote printer at location $J.#IBM.DEFAULT:

```
ENFORM/IN DAILYRPT, OUT $J.#IBM.DEFAULT/
```

Chapter 9: Operation and Control

This chapter explains the control and monitoring of CA XCOM Data Transport from an HP NonStop workstation as they relate to:

- Log files and trace files
- The history of a transfer
- Event management
- Checkpoint/restart
- NetBatch
- Running CA XCOM Data Transport in the background

This section contains the following topics:

Log Files and Trace Files

Logging Locally Initiated Transfers

The XLOGFILE parameter specifies which log file CA XCOM Data Transport will use to keep a history of all locally initiated transfers, log their status, and record various errors that may occur. The XLOGFILE parameter default is XCOMLOG.

The local trace file is directed to STDERR. To redirect it to a file, use the following:

```
ASSIGN STDERR,filename
```

Logging Remotely Initiated Transfers

For remote transfers, CA XCOM Data Transport stores the remote log file and remote trace file based on the parameter values of RLOGFILE and RTRACEFILE in the xcomcnf file.

The remote log file name is in this format:

RL*yymmdd*

The remote trace file name is in this format:

RT*yymmdd*

The parameter XTRACE determines how much trace information is written to the trace file.

Access Transfer History

CA XCOM Data Transport for HP NonStop maintains a record of all active and restarted transfers in a history file. The default name of this history file is XCOMHIST. You can display this file on your screen using a program called SCANHIST.

View XCOMHIST

To view XCOMHIST

Type the following command:

vol.subvol.scanhist *vol.subvol*.xcomhist

The system displays the following:

```
[D]isplay one detailed record, [L]ist Summary, [P]urge, [Q]uit? [DLQP]
```

You can request information on a single transfer, or sort the information by request number or date.

Display a Specific Record

To display a specific record

1. Type D and press ENTER.

   You are prompted for the request number of the record.

2. Type the request number of the record you want to see and press ENTER.

   The requested transfer record detail is displayed, as in the following example:

```
REQUEST NO : 000011          TRANSFER ID : MY_TRANS
TYPE : LOCAL SEND FILE
STATUS : A ERROR : 310
LOCAL LU  : LUICE
REMOTE LU : LUSTRATT
LOCAL FILE  : CEXTDECS
REMOTE FILE : xcom2>tester
XCOM LOGIN   : GREEN
XCOM CREATOR : SCI MANAGER
NOTIFY USER  : GREEN
TRANSFER STARTED : 1993/04/26, 9:47:57
TRANSFER TIME    : 0
RECORDS TRANSFERRED : 0
BYTES TRANSFERRED   : 0
COMPRESSED BYTES TRANSFERRED : 0
-- [D]isplay one detailed record, [L]ist Summary, [P]urge, [Q]uit? [DLQP]
```

Display a Summary List

To display a summary list

1. Type **L** and press ENTER.

   You are prompted for a sort value.

2. Use the following list to decide on your next step:

   - If you want to sort by date, then go to Step 3.
   - If you want to sort by record number, then go to Step 5.

3. Type **D** and press ENTER.

   You are prompted for a date.

4. Type in a date in the following format and press ENTER:

   `MM/DD/YY`

   The specified record summary is displayed.

   Note: If you press ENTER without entering a value, the default is used.

5. Type **R** and press ENTER.

   The specified record summary is displayed.

   Note: If you press ENTER without entering a value, the default is used.

Sample Record Summary List

The following is a sample summary list of records sorted by date:

```
REQUEST NO   L/R  S/R F/J/R STAT  ERR    STARTED               TIME    RECS KBYTES
000002        L    S    F    A    310    1993/05/20, 11:28:03   0       0    0
000002        L    S    F    R    310    1993/05/20, 11:28:04   0       0    0
000004        L    S    F    A    310    1993/05/20, 11:33:50   0       0    0
000004        L    S    F    R    310    1993/05/20, 11:33:52   0       0    0
000367        R    R    F    X    403    1993/05/20, 14:12:24   0       0    0
000367        R    R    F    X    403    1993/05/20, 14:12:25   0       0    0
000382        R    R    F    V    412    1993/05/20, 15:24:36   1626    80   4
No more records found
-- [D]isplay one detailed record, [L]ist Summary, [P]urge, [Q]uit? [DLQP]
```

Purge Records

To purge records

1. Type **P** and press ENTER.

   You are prompted to pause every 20 lines. Answer Y or N.

   You are prompted for a sort value.

2. Use the following list to decide your next step:

   - If you want to sort by date, then go to Step 3.
   - If you want to sort by request number, then go to Step 6.

3. Type **D** and press ENTER.

   You are prompted for a FROM date.

4. Type a FROM date in the following format and press ENTER:

   `MM/DD/YY`

   You are prompted for a TO date.

   Note: If you press ENTER without entering a value, the default is used.

5. Enter a TO date in the following format and press ENTER:

   `MM/DD/YY`

   A list of deleted records that you requested to be purged is displayed.

   Note: If you press ENTER without entering a value, the default is used.

6. Type **R** and press ENTER.

   You are prompted for a FROM request #.

7. Type a valid FROM request # and press ENTER.

   You are prompted for a TO request #.

   Note: If you press ENTER without entering a value, the default is used.

8. Type a valid TO request # and press ENTER.

   A list of deleted records that you requested to be purged is displayed.

Sample Purged Records List

The following is a sample purged records list sorted by date:

```
Purging    Request  #000256      2011/05/01     10:11:55
Purging    Request  #000257      2011/05/01     10:30:45
Purging    Request  #000258      2011/05/01     10:35:50
Purging    Request  #000259      2011/05/01     10:38:20
```

Record Summary List Fields

The record summary list fields are as follows:

REQUEST NO

The request number.

L/R

Indicates if the transfer is locally or remotely initiated.

S/R

Indicates whether the transfer was a send or receive request.

F/J/R

Indicates whether a file, job, or report was transferred.

STAT

Indicates the transfer status.

A

An active locally initiated transfer.

C

A completed transfer.

I

A transfer that was incomplete.

R

A locally initiated transfer that requires restart.

X

An active remotely initiated transfer.

V

A remotely initiated transfer that requires restart.

1

A Version 1 transfer.

ERR

The error number (if any).

STARTED

The transfer's start date and time.

TIME

The amount of time, in seconds, it took to complete the transfer.

RECS

How many records were transferred.

KBYTES

How many Kbytes were transferred (rounded to the nearest KB).

Event Management Service (EMS)

CA XCOM Data Transport for HP NonStop uses the Event Management Service (EMS) to monitor CA XCOM Data Transport events and to write user exits. EMS event messages relay CA XCOM Data Transport file transfer information to the user or application program.

The following CA XCOM Data Transport for HP NonStop events generate messages:

- The start of a transfer
- The end of a transfer
- The ABEND of a transfer

CA XCOM Data Transport for HP NonStop sends event messages to an EMS collector that writes them to a system log in EMS format. In addition to the event number and message text, an EMS message includes the following:

- A request number
- A transfer ID
- Whether the transfer is a file, job, or report
- Whether it is a send or receive transfer

The following pages describe the EMS tokens, explain how to use filters to access CA XCOM Data Transport events, and show how to generate EMS reports.

Note: For more information about EMS, see the HP NonStop *Event Management Services (EMS) Guide*.

EMS Tokens

There are two kinds of CA XCOM Data Transport for HP NonStop event messages: critical and informative. Critical messages indicate an error that usually requires some action, while informative messages convey information but do not require any action.

All event messages are made up of EMS tokens. EMS messages have a set of standard and product specific tokens. All of the tokens required for each CA XCOM Data Transport for HP NonStop event are contained in the following supplied files:

- XCM1C (C language)
- XCM1TAL (TAL)
- XCM1COB (COBOL)
- XCM1TACL (TACL)

Use the definitions found in the file for the language of your program.

EMS Filters

You can write EMS filters to select messages based on the values of the fields, or tokens, in the messages. For example, you can use the provided sample filter file XCM1EMFS with EMSDIST or VIEWPOINT to display all CA XCOM Data Transport EMS messages.

You can also write filters that access only selected CA XCOM Data Transport EMS messages such as:

- All completed transfers
- All aborted transfers
- All started transfers

EMS filters are very useful for application programmers. For example, a filter can be used for an application program that is waiting to receive a certain file. When the file is received, the program receives the EMS message from CA XCOM Data Transport and can then process the file.

To allow CA XCOM Data Transport to write messages to EMS, you need to add the following DEFINEs:

```
ADD DEFINE =_EMS_COLLECTOR, class map, file $0
```

This line defines the EMS collector for CA XCOM Data Transport as $0, which is the default collector. If needed, you can set this value to another collector:

```
ADD DEFINE =_EGEN_ADD_EVENT_TEXT, class map, file $YES
```

This define adds a text token to CA XCOM Data Transport EMS messages. If you are printing messages, EMSTEXT generates displayable message text from this token.

Note: The CA XCOM Data Transport EMS messages do not include a format template.

For more information about writing your own filters and programs that communicate with EMS, see the *HP NonStop Event Management System (EMS) Guide.*

Token Format

The supplied files XCM1C, XCM1TAL, XCM1COB, and XCMITACL contain the definitions of the CA XCOM Data Transport EMS tokens. Include these files with your application program to specify the correct format and values for the CA XCOM Data Transport EMS tokens.

The following is an example of the XCM1C file:

```
/* SCHEMA PRODUCED DATE - TIME : 5/11/93 14:10:57                              */
#pragma section zspi_ddl_char254
/*-----------------------------------------------------------------------------*/
/*                                                                             */
/*  EMS FastStart - T9263C20 - (17MAR91)                                       */
/*                                                                             */
/*  DDL Source Library Files, Language Dependant                               */
/*                                                                             */
/*  Produced by the DDL compilation of XCM1ddls                                */
/*                                                                             */
/*  Generation Time:   May 11, 1993 13:51:18                                   */
/*                                                                             */
/*-----------------------------------------------------------------------------*/
/*-----------------------------------------------------------------------------*/
/* Source the EXTRADDL source schema file for user defined SPI data types.     */
/*-----------------------------------------------------------------------------*/
/*-----------------------------------------------------------------------------*/
/*                                                                             */
/*  EMS FastStart - T9263C20 - (17MAR91)                                       */
/*                                                                             */
/*  File Type:                    DDL Source Schema                            */
/*                                                                             */
/*  Source File Name:             Extraddl                                     */
/*                                                                             */
/*  Generation Time:              July 7, 1988                                 */
/*                                                                             */
/*  Language Compiler Required:   Data Definition Language (DDL)               */
/*                                                                             */
/*  Compiler Version Required:    C20                                          */
/*                                                                             */
/*  Source Libary File Produced:  Sourced during the compilation of            */
/*                                the main DDL file.                           */
/*                                                                             */
/*  File Description: This DDL source schema file is an example of DDL         */
/*  definitions which may be added to the base ZSPIDDL definitions             */
/*  provided by Tandem.  These definitions can then be used by                 */
/*  EMS FastStart and EGEN to create tokens of specific types.                 */
/*                                                                             */
/*  Modifications Summary:                          Date of Modification       */
/*                                                                             */
```

```
/*  1- Added the Zspi-ddl-char254 token.  Used by       21 October, 1988      */
/*     EGEN to generate an event message with a                               */
/*     ZEMS-TKN-TEXT of up to 254 bytes.                                      */
/*     N.B. 254 is the maximum byte length for a                              */
/*          fixed token code.                                                 */

/*                                                                            */
/*----------------------------------------------------------------------------*/
typedef struct
{
   union
   {
      char                          z_c[254];
      struct
      {
         short                      z_i[127];
      } z_s;
      char                          z_b[254];
   } u_z_c;
} zspi_ddl_char254_def;
#pragma section zspi_typ_char254
#define ZSPI_TYP_CHAR254 510u
#pragma section zspi_ddl_char10
typedef struct
{
   union
   {
      char                          z_c[10];
      struct
      {
         short                      z_i[5];
      } z_s;
      char                          z_b[10];
   } u_z_c;
} zspi_ddl_char10_def;
#pragma section zspi_typ_char10
#define ZSPI_TYP_CHAR10 266U
#pragma section xcom_ssid
/*----------------------------------------------------------------------------*/
/*                                                                            */
/* XCOM SSID is defined here and will be passed to the EGEN                   */
/* procedure to identify the owner of the event.  The SSID definition         */
/* will also be used by EMF to compile the FILTER example.                    */
/*                                                                            */
/*   Description          Value                                               */
/*   -----------          -----                                               */
/*                                                                            */
/*  XCOM-VAL-OWNER:         XCOM                                              */
/*                                                                            */
```

```
/*  XCOM-SSN-NUMBER:        1                                                 */
/*                                                                            */
/*  XCOM-VAL-VERSION:       V02                                               */
/*                                                                            */
/*----------------------------------------------------------------------------*/
/*                                                                            */
/*                                                                            */
/*                                                                            */
#define XCOM_VAL_OWNER "XCOM"
#define XCOM_SSN_NUMBER 1
#define XCOM_VAL_VERSION 22018u
/*                                                                            */
/*                                                                            */
/*                                                                            */
typedef struct
{
   union
   {
      char                          z_filler[8];
      /*value is "XCOM"*/
      zspi_ddl_char8_def            z_owner;
   } u_z_filler;
   zspi_ddl_int_def              z_number;
   /*value is 1*/
   zspi_ddl_uint_def             z_version;
   /*value is 22018*/
} xcom_val_ssid_def;
#pragma section egen_record
typedef struct
{
   zspi_ddl_int_def              acf_version;
   zspi_ddl_char8_def            ssid_owner;
   zspi_ddl_int_def              ssid_subsystem_number;
   zspi_ddl_uint_def             ssid_version;
   zspi_ddl_int_def              egen_error;
   zspi_ddl_enum_def             event_type;
   zspi_ddl_int_def              event_number;
   zspi_ddl_int_def              action_id;
   zspi_ddl_boolean_def          suppress_display;
   zspi_ddl_fname32_def          subsystem_manager;
   zspi_ddl_char254_def          event_text;
   zspi_ddl_char24_def           subject_field_name;
   zspi_ddl_enum_def             msg_number;
   zspi_ddl_enum_def             filetype;
   zspi_ddl_fname_def            local_filename;
   zspi_ddl_char64_def           remote_filename;
   zspi_ddl_char8_def            local_luname;
   zspi_ddl_char8_def            remote_luname;
   zspi_ddl_int2_def             record_count;
```

```
   zspi_ddl_int2_def              byte_count;
   zspi_ddl_int4_def              micro_seconds;
   zspi_ddl_int_def               compress_svngs;
   zspi_ddl_int_def               spool_job_num;
   zspi_ddl_int_def               error_code;
   zspi_ddl_int_def               sub_error_code;
   zspi_ddl_enum_def              local_remote;
   zspi_ddl_char254_def           error_text;
   zspi_ddl_char10_def            transfer_id;
   zspi_ddl_char6_def             request_no;
} egen_record_def;
#pragma section egen_interface_definitions
/*-------------------------------------------------------------------------------*/
/* EGEN module interface variables definitions.                                  */
/*-------------------------------------------------------------------------------*/
/*                                                                               */
/* Used to define the EVENT-TYPE field of EGEN-RECORD.                           */
/*                                                                               */
#define INFORMATIVE_EVENT 1
#define ACTION_ATTENTION_EVENT 2
#define ACTION_COMPLETION_EVENT 3
#define CRITICAL_EVENT 4
#define ACF_VERSION 16896u
/**/
/*-------------------------------------------------------------------------------*/
/* When the EGEN module is invoked, it will return a status variable,            */
/* called Return-code, to the calling program.  Please note that this            */
/* return code does not correspond to a file system error, the field             */
/* egen-error of egen-record will contain more information. This                 */
/* table lists the possible values returned by EGEN.                             */
/*                                                                               */
/* Return-Code Description                                                       */
/*                                                                               */
/*      0      EGEN successfully generated the event message                     */
/*                                                                               */
/*      1      The Initialize^egen^record procedure detected that                */
/*             there was no parameter passed to this procedure.                  */
/*                                                                               */
/*     10-13   The Open^egen^collector procedure detected an error               */
/*             when opening the collector.  The Egen-error field                 */
/*             of Egen-record contains detailed information.                     */
/*                                                                               */
/*     20-21   The Close^egen^collector procedure detected an error              */
/*             when closing the collector.  The Egen-error field                 */
/*             of Egen-record contains detailed information.                     */
/*                                                                               */
/*     30-31   The Complete^egen^operation procedure detected an error           */
/*             when completing the write operation. The Egen-error field         */
/*             of Egen-record contains detailed information.                     */
```

```
/*                                                                          */
/*     40-41   The Get^egen^event^text^define procedure detected an error   */
/*             when processing the =_EGEN_ADD_EVENT_TEXT define.  The       */
/*             Egen-error field of Egen-record contains detailed            */
/*             information.                                                 */
/*                                                                          */
/*     50-59   The Initialize^event^buffer procedure detected an error      */
/*             when initializing the event buffer.  The Egen-error field    */
/*             of Egen-record contains detailed information.                */
/*                                                                          */
/*     60-61   The Write^event^buffer procedure detected an error or a warning */
/*             when writing the event buffer.  The Egen-error field         */
/*             of Egen-record contains detailed information.                */
/*                                                                          */
/*     70      The Egen procedure detected that a required parameter        */
/*             was not passed or that the combination of parameters was     */
/*             not valid.                                                   */
/*                                                                          */
/*     71      The Egen procedure detected that the Egen-record was not     */
/*             initialized by the Initialize^egen^record procedure          */
/*             before calling Egen.                                         */
/*                                                                          */
/*--------------------------------------------------------------------------*/
/*                                                                          */
/* Constants and returns code used by the Initialize^egen^record procedure  */
/*                                                                          */
#define EGEN_INITIALIZE_RECORD_OK 0
#define EGEN_INITIALIZE_MISSING_PARAM 1
/*                                                                          */
/* Constants and returns code used by the Open^egen^collector procedure     */
/*                                                                          */
#define EGEN_OPEN_COLLECTOR_OK 0
#define EGEN_OPEN_MISSING_PARAM 10
#define EGEN_OPEN_INVALID_SYNC_DEPTH 11
#define EGEN_OPEN_COLLECTOR_ERROR 12
#define EGEN_OPEN_COLLECTOR_WARNING 13
/*                                                                          */
/* Constants and returns code used by the Close^egen^collector procedure    */
/*                                                                          */
#define EGEN_COLLECTOR_CLOSED_OK 0
#define EGEN_COLLECTOR_MISSING_PARAM 20
#define EGEN_COLLECTOR_ALREADY_CLOSED 21
/*                                                                          */
/* Constants and returns code used by the Complete^egen^operation procedure */
/*                                                                          */
#define EGEN_COMPLETE_OPERATION_OK 0
#define EGEN_COMPLETE_MISSING_PARAM 30
#define EGEN_COMPLETE_OPERATION_ERROR 31
/*                                                                          */
```

```
/* Constants and returns code used by the Get^egen^event^text^define procedure */
/*                                                                              */
#define EGEN_GET_TEXT_DEFINE_OK 0
#define EGEN_GET_TEXT_DEFINEMODE_ERROR 40
#define EGEN_GET_TEXT_DEFINEINFO_ERROR 41
/*                                                                              */
/* Constants and returns code used by the Initialize^event^buffer procedure    */
/*                                                                              */
#define EGEN_INITIALIZE_EVENT_OK 0
#define EGEN_INITIALIZE_TYPE_ERROR 50
#define EGEN_INITIALIZE_EVENT_NUMBER 51
#define EGEN_INITIALIZE_EMSINIT_ERROR 52
#define EGEN_INITIALIZE_SUBJECT_ERROR 53
#define EGEN_INITIALIZE_FLAGS_ERROR 54
#define EGEN_INITIALIZE_ACTION_ID 55
#define EGEN_INITIALIZE_ACTION_ERROR 56
#define EGEN_INITIALIZE_TEXT_ERROR 57
#define EGEN_INITIALIZE_TOKENS_ERROR 58
#define EGEN_INITIALIZE_SSGETTKN_ERROR 59
/*                                                                              */
/* Constants and returns code used by the Write^event^buffer procedure         */
/*                                                                              */
#define EGEN_WRITE_EVENT_OK 0
#define EGEN_WRITE_EVENT_WARNING 60
#define EGEN_WRITE_EVENT_ERROR 61
/*                                                                              */
/* Constants and returns code used by the Egen procedure                        */
/*                                                                              */
#define EGEN_GENERATE_EVENT_OK 0
#define EGEN_MISSING_PARAMETER_ERROR 70
#define EGEN_RECORD_NOT_INITIALIZED 71
/*------------------------------------------------------------------------    --*/
/* We turn off the creation of COBOL DDL since the next sections                */
/* will only be used by EGEN or by the TACL filter language.                    */
/*------------------------------------------------------------------------------*/
/*                                                                              */
/* Constants used by Egen to define the default values of a field               */
/*                                                                              */
#define EMSFS_DEFAULT_INT 32767
#define EMSFS_DEFAULT_INT2 2147483647
#define EMSFS_DEFAULT_INT4 9223372036854775807
#define EMSFS_DEFAULT_TRANSID 9223372036854775807
#define EMSFS_DEFAULT_TIMESTAMP 9223372036854775807
#define EMSFS_DEFAULT_UINT 65535
#define EMSFS_DEFAULT_ENUM 32767
#pragma section xcom_token_definitions
/*------------------------------------------------------------------------------*/
/* The application tokens are defined here and correspond one to one            */
/* with the application fields defined in the ACF.                              */
```

```
/*                                                                          */
/* The token code is build by using the field name index as the token       */
/* number and the field type as the token type.                             */
/*--------------------------------------------------------------------------*/
#define XCOM_TNM_MSG_NUMBER 100
#define XCOM_TKN_MSG_NUMBER 184680548lu
#define XCOM_TNM_FILETYPE 200
#define XCOM_TKN_FILETYPE 184680648lu
#define XCOM_TNM_LOCAL_FILENAME 300
#define XCOM_TKN_LOCAL_FILENAME 337117484lu
#define XCOM_TNM_REMOTE_FILENAME 400
#define XCOM_TKN_REMOTE_FILENAME 20971920lu
#define XCOM_TNM_LOCAL_LUNAME 500
#define XCOM_TKN_LOCAL_LUNAME 17302004lu
#define XCOM_TNM_REMOTE_LUNAME 600
#define XCOM_TKN_REMOTE_LUNAME 17302104lu
#define XCOM_TNM_RECORD_COUNT 700
#define XCOM_TKN_RECORD_COUNT 50594492lu
#define XCOM_TNM_BYTE_COUNT 800
#define XCOM_TKN_BYTE_COUNT 50594592lu
#define XCOM_TNM_MICRO_SECONDS 900
#define XCOM_TKN_MICRO_SECONDS 67634052lu
#define XCOM_TNM_COMPRESS_SVNGS 1000
#define XCOM_TKN_COMPRESS_SVNGS 33686504lu
#define XCOM_TNM_SPOOL_JOB_NUM 1100
#define XCOM_TKN_SPOOL_JOB_NUM 33686604lu
#define XCOM_TNM_ERROR_CODE 1200
#define XCOM_TKN_ERROR_CODE 33686704lu
#define XCOM_TNM_SUB_ERROR_CODE 1300
#define XCOM_TKN_SUB_ERROR_CODE 33686804lu
#define XCOM_TNM_LOCAL_REMOTE 1400
#define XCOM_TKN_LOCAL_REMOTE 184681848lu
#define XCOM_TNM_ERROR_TEXT 1500
#define XCOM_TKN_ERROR_TEXT 33424860lu
#define XCOM_TKN_TRANSFER_ID 1600
#define XCOM_TKN_TRANSFER_ID 17172032lu
#define XCOM_TKN_REQUEST_NO 1700
#define XCOM_TKN_REQUEST_NO 17172132lu
#pragma section egen_subject_token
/*--------------------------------------------------------------------------*/
/* The EGEN-SUBJECT-TOKEN definition is used by the procedure               */
/* Initialize^event^buffer within the EGEN module to add the                */
/* subject-name as the subject token.  This is only used by the             */
/* EGENPROG test program to have a default subject since we do not          */
/* know what fields the user will define in it's ACF.                       */
/*--------------------------------------------------------------------------*/
#define XCOM_TNM_SUBJECT_NAME 9998
#define XCOM_TKN_SUBJECT_NAME 18360078lu
#pragma section xcom_event_numbers
```

```
/*---------------------------------------------------------------------------*/
/* If a file name was specified with the USER-DDL-FILE key word              */
/* in the ACF, it will be sourced here.  These definitions are also          */
/* added in all the source library files (C, COBOL85, TACL and TAL).         */
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
#define XCOM_FILE_SEND_STARTING 99
#define XCOM_REPORT_SEND_STARTING 98
#define XCOM_JOB_SEND_STARTING 97
#define XCOM_FILE_RECEIVE_STARTING 96
#define XCOM_REPORT_RECEIVE_STARTING 95
#define XCOM_JOB_RECEIVE_STARTING 94
#define XCOM_FILE_SEND_ENDED 93
#define XCOM_REPORT_SEND_ENDED 92
#define XCOM_JOB_SEND_ENDED 91
#define XCOM_FILE_RECEIVE_ENDED 90
#define XCOM_REPORT_RECEIVE_ENDED 89
#define XCOM_JOB_RECEIVE_ENDED 88
#define XCOM_FILE_SEND_ABORTED 87
#define XCOM_REPORT_SEND_ABORTED 86
#define XCOM_JOB_SEND_ABORTED 85
#define XCOM_FILE_RECEIVE_ABORTED 84
#define XCOM_REPORT_RECEIVE_ABORTED 83
#define XCOM_JOB_RECEIVE_ABORTED 82
#pragma section xcom_enum_values
/* Values for xcom-local-remote token                                        */
#define XCOM_LOCAL 1
#define XCOM_REMOTE 2
/* Values for xcom-filetype token                                            */
#define XCOM_FILETYPE_JOB 1
#define XCOM_FILETYPE_REPORT 2
#define XCOM_FILETYPE_FILE 3
```

Tokens Common to All Events

The following tokens are common to all events:

XCOM-TKN-FILETYPE

Indicates the type of transferred file. Depending on the CA XCOM Data Transport for HP NonStop command issued, this value can be any of the following:

- XCOM-FILETYPE-JOB
- XCOM-FILETYPE-REPORT
- XCOM-FILETYPE-FILE

XCOM-TKN-LOCAL-FILENAME

Contains the name of the file on the local HP NonStop system. If receiving a report, this is the spool collector location name.

XCOM-TKN-LOCAL-LUNAME

Indicates the name of the local SNA LU as specified by the XLUNAME parameter.

XCOM-TKN-LOCAL-REMOTE

Indicates the location where the transfer was initiated (either XCOM-LOCAL or XCOM-REMOTE).

XCOM-TKN-MSG-NUMBER

Contains the CA XCOM Data Transport message number. Each CA XCOM Data Transport for HP NonStop message is assigned a message number by CA XCOM Data Transport, whether it is written to a terminal, the CA XCOM Data Transport log file, a remote CA XCOM Data Transport process, or the HP NonStop event log.

For a list of message numbers and text, see the appendix "Messages."

XCOM-TKN-REMOTE-FILENAME

Indicates the name of the remote file associated with the transfer.

XCOM-TKN-REMOTE-LUNAME

Indicates the name of the remote SNA LU specified by the REMOTE_SYSTEM parameter.

XCOM-TKN-REQUEST-NO

Indicates the ID number of each transfer.

XCOM-TKN-SUBJECT-NAME

Identifies the subject. This value is always "*local-filename*."

XCOM-TKN-TRANSFER-ID

Specifies the non-unique user-assigned identifier for each transfer.

ZSPI-TKN-SSID

This is the subsystem ID. To check for CA XCOM Data Transport messages, check for the following:

ZSPI^TKN^SSID=XCOM^VAL^SSID

The token XCOM^VAL^SSID is composed of three other tokens:

- XCOM-VAL-OWNER=XCOM
- XCOM-SSN-NUMBER=1
- XCOM-VAL-VERSION=VO2

ZEMS-TKN-EMPHASIS

Indicates if the event is critical or informative. For CA XCOM Data Transport, the event is critical only if it is an aborted transfer.

ZEMS-TKN-EVENTNUMBER

Specifies the number assigned to event.

ZEMS-TKN-TEXT

Contains the message text.

Tokens Common to the Successful Completion of a Transfer

The following tokens are common to the successful completion of a transfer:

XCOM-TKN-BYTE-COUNT

Specifies the number of bytes transferred. Used only in messages for completed transfers.

XCOM-TKN-COMPRESS-SVNGS

Specifies the percentage of total characters saved by compression.

XCOM-TKN-MICRO-SECONDS

Specifies the time in microseconds required for the transfer to complete.

XCOM-TKN-RECORD-COUNT

Specifies the number of records transferred. Used only in messages for completed transfer.

XCOM-TKN-SPOOL-JOB-NUMS

Specifies the job number for received reports as known to the spooler.

Tokens Common to Aborted Transfers

The following tokens are common to aborted transfers:

XCOM-TKN-ERROR-CODE

Specifies a CA XCOM Data Transport Guardian error code.

XCOM-TKN-ERROR-SUBCODE

Specifies the CA XCOM Data Transport Guardian error subcode.

XCOM-TKN-ERROR-TEXT

Specifies the text of a locally or remotely generated error message.

Sample Event Token File

The following sample event token file, EMSFSDDL, includes the token values for the event numbers, the local-remote flag, and the file types used for CA XCOM Data Transport for HP NonStop EMS messages:

```
?PAGE "CA-XCOM EMS FastStart DDL schema source file"
*-----------------------------------------------------
?SETSECTION XCOM-event-numbers
CONSTANT XCOM-file-send-starting            VALUE IS 99.
CONSTANT XCOM-report-send-starting          VALUE IS 98.
CONSTANT XCOM-job-send-starting             VALUE IS 97.
CONSTANT XCOM-file-receive-starting         VALUE IS 96.
CONSTANT XCOM-report-receive-starting       VALUE IS 95.
CONSTANT XCOM-job-receive-starting          VALUE IS 94.
CONSTANT XCOM-file-send-ended               VALUE IS 93.
CONSTANT XCOM-report-send-ended             VALUE IS 92.
CONSTANT XCOM-job-send-ended                VALUE IS 91.
CONSTANT XCOM-file-receive-ended            VALUE IS 90.
CONSTANT XCOM-report-receive-ended          VALUE IS 89.
CONSTANT XCOM-job-receive-ended             VALUE IS 88.
CONSTANT XCOM-file-send-aborted             VALUE IS 87.
CONSTANT XCOM-report-send-aborted           VALUE IS 86.
CONSTANT XCOM-job-send-aborted              VALUE IS 85.
CONSTANT XCOM-file-receive-aborted          VALUE IS 84.
CONSTANT XCOM-report-receive-aborted        VALUE IS 83.
CONSTANT XCOM-job-receive-aborted           VALUE IS 82.
?SETSECTION XCOM-enum-values
* Values for xcom-local-remote token
CONSTANT XCOM-local                         VALUE IS 1.
CONSTANT XCOM-remote                        VALUE IS 2.
* Values for xcom-filetype token
CONSTANT XCOM-filetype-job                  VALUE IS 1.
CONSTANT XCOM-filetype-report               VALUE IS 2.
CONSTANT XCOM-filetype-file                 VALUE IS 3.
```

Using EMS Filters to Access EMS Events

Use the following excerpt from the XCM1EMFS file to select all messages for the CA XCOM Data Transport subsystem. To select a subset of these messages, rewrite this example and use the EMF compiler.

```
------------------------------------------------------------------
[#SET ZEMS^VAL^SSID [ZSPI^VAL^TANDEM].[ZSPI^SSN^ZEMS].0]
[#SET XCOM^VAL^SSID [XCOM^VAL^OWNER].[XCOM^SSN^NUMBER].0]
FILTER XCOM^DEFAULT^FILTER;
BEGIN SSID ( ZEMS^VAL^SSID )
  IF ZSPI^TKN^SSID = SSID ( XCOM^VAL^SSID ) THEN
     BEGIN
        --
        -- Fails on suppress^display events which are not
        -- action-completion.
        --
        IF ZEMS^TKN^SUPPRESS^DISPLAY = [ZSPI^VAL^TRUE] THEN
        BEGIN
          IF TOKENPRESENT ( ZEMS^TKN^ACTION^NEEDED ) AND
             ZEMS^TKN^ACTION^NEEDED = [ZSPI^VAL^FALSE] THEN PASS 3
          ELSE
            FAIL;
        END;
        --
        -- Passes action-attention and action-completion events
        --
        IF TOKENPRESENT ( ZEMS^TKN^ACTION^NEEDED ) THEN PASS 1;
        --
        --
        -- Testing for  false for critical events
        --
        IF ZEMS^TKN^EMPHASIS  [ZSPI^VAL^FALSE] THEN PASS 2;
        --
        --
        -- All other events from XCOM^VAL^SSID are passed
        --
        PASS;
     END
  ELSE FAIL;
END;
```

Sample EMS Report

Two files, EMSVIEW and EMSRVIEW, provide examples of reporting EMS messages. Both files run EMSDIST with the XCM1EMFO filter. EMSCVIEW creates a process that reports all new CA XCOM Data Transport messages; EMSRVIEW reviews old CA XCOM Data Transport messages.

The following is a sample EMS report that was obtained by running EMSRVIEW:

```
$CLX12 SCI 69>EMSRVIEW
92-11-20 13:44:27 \NETPTC1.$Z469    CA-XCOM.1.V02        000096 XCOMT0014I
                                    Receiving local file
                                    \NETPTC1.$clx12.sci.delete7 from XCOMQA
92-11-20 13:44:36 \NETPTC1.$Z469    CA-XCOM.1.V02        000090 XCOMT0002I
                                    Received local file
                                    \NETPTC1.$CLX12.SCI.DELETE7 from XCOMQA ;
                                    5 records, 400 bytes, in 0 seconds (1422
                                    bytes/sec).
```

The tokens for the first message are as follows:

SSID token

CA-XCOM.1.VO2

Event token

96

Event text

The rest of the message

Except for the event token, the second message uses the same tokens:

SSID token

CA-XCOM.1.VO2

Event token

90

Event text

The rest of the message

To include other tokens in your messages, you must specify them in your filter.

Checkpoint/Restart

Checkpoint/restart is a major feature of all CA XCOM Data Transport Version 2 products. Checkpoint/restart allows you to restart a failed transfer from the last confirmed checkpoint so you do not have to requeue a failed transfer from the beginning.

When a transfer starts, a record of the transfer is created and stored in a checkpoint file whose name was specified in the CHECKPOINT_FILE parameter. Every time the number of records specified in the checkpoint count successfully transfers, the checkpoint file is updated. If the entire file is transferred successfully, the record of that transfer is deleted from the checkpoint file. If the transfer fails, it can be restarted from the last checkpoint.

This feature is useful when transferring data over SDLC dial-out lines, where the quality of the switched connection cannot be guaranteed. When using a reliable connection, like Token-Ring, or Ethernet LAN, or a channel connection using SNAXLink, the checkpoint/restart feature should be disabled by specifying CHECKPOINT_COUNT=0. The checkpoint/restart feature adds overhead that degrades performance.

Specifying a Checkpoint

There are two ways to specify the checkpoint count and checkpoint file:

- Using the command line
- Using the XCOMCNF configuration file

Using the Command Line

To specify a checkpoint using the command line

Type the following on the command line:

```
run XCOM62 put myfile as theirfil, checkpoint_count=checkpoint_count,
checkpoint_file=checkpoint_filename, retries=1, restart_supported=yes
```

Notes:

- Make sure you type the entire command before pressing ENTER, or you will execute an incomplete command.
- Also make sure that restart_supported=YES and that the retries=*value* parameter is greater than zero. Otherwise, the transfer cannot be restarted.

Using the XCOMCNF Configuration File

To specify a checkpoint using the XCOMCNF configuration file

Specify the desired checkpoint count and checkpoint file in the CHECKPOINT_COUNT and CHECKPOINT_FILE parameters in the XCOMCNF configuration file.

Restart a Failed Transfer

When a transfer is restarted, the local and remote CA XCOM Data Transport programs must agree where the transfer should restart. There are two ways to restart a transfer:

- Using the command line
- Using XCOMDMN

Using the Command Line

To restart a transfer using the command line

Specify the request number and the remote LU name, and set the RESTART_FLAG parameter to YES. CA XCOM Data Transport for HP NonStop tries to look up the transfer record in the checkpoint file and restart it.

Note: When specifying the request number during a restart from the command line, make sure that the request number is six characters long and is padded with zeros, for example, REQUEST_NO=000012.

To force a manual restart

Type the following on the command line:

```
run XCOM62 put myfile as theirfil, request_no=request_#,
remote_system=remote_LU_name, restart_flag=YES
```

Note: Make sure you type the entire command before pressing ENTER, or you will execute an incomplete command.

Important! There is a tradeoff between performance and frequency of checkpoints. Such factors as transmission line quality and file size determine how often you should checkpoint a particular transfer.

Using XCOMDMN

You can restart a transfer using XCOMDMN, the CA XCOM Data Transport daemon process on the Tandem platform.

To restart a transfer using XCOMDMN

1. Issue the following command to define the location of CA XCOM Data Transport to the daemon:

   `ADD DEFINE =XCOM62-PROGRAM, CLASS MAP, FILE vol.subvol.XCOM62`

   XCOMDMN periodically reads the checkpoint file, searching for failed transfers that need to be restarted. XCOMDMN is started once and runs indefinitely.

2. To execute the daemon, issue the following command:

   `run vol.subvol.xcomdmn /nowait, out listfile/vol.subvol.checkpoint_filename`

   Note: Make sure you type the entire command before pressing ENTER, or you will execute an incomplete command.

About XCOMDMN

XCOMDMN handles the automatic retry of failed transfers and the scheduling of transfers. It performs some of the same functions as the CA XCOM Data Transport daemon on UNIX and NT, but it is quite different.

The following sections describe the parameters for XCOMDMN.

RETRIES

Indicates the number of times a transfer should be retried by the daemon.

Note: A transfer cannot be resumed or restarted unless this parameter is set to a value greater than zero.

Range: 0 to 32767

Default: 0

RETRY_TIME

Indicates number of seconds the daemon program should wait for a transfer to be started.

Range: 0 to 32767

Default: 60 seconds

RECYCLE (Tandem Parameter)

Indicates how many seconds the daemon program should wait between scans of the checkpoint file.

Note: Be aware of the settings of both the RECYCLE and RETRY_TIME parameters. For example, if the RETRY-TIME is set to 60 seconds, but the RECYCLE parameter is set to 300 seconds, then the transfer does not restart for at least 60 seconds. However, it may be 300 seconds until the daemon program actually reads the checkpoint file and restarts the transfer.

Default: 300 seconds (five minutes)

REMOTE_EXPIRE (HP NonStop Parameter)

Specifies the number of seconds the daemon program should wait before purging remotely initiated transfers marked for restart (status R displayed by XCOMQM).

Default: 300 seconds (five minutes)

Example:

The RECYCLE and REMOTE_EXPIRE HP NonStop parameters can be used as shown in the following example:

```
run vol.subvol.xcomdmn /nowait, out
listfil/vol.subvol.ckptfil,recycle=500,remote_expire=500
```

REQUEST_NO

System-generated.

Specifies the unique ID number associated with each transfer.

Range: Up to six characters, with leading zeros required for numbers less than six characters (for example, 000034).

Default: None

RESTART_FLAG

Indicates if a transfer is a restart request. This parameter can be used to force a restart.

Range: YES or NO

Default: NO

RESTART_SUPPORTED

Determines whether a locally initiated transfer can be started, as follows:

- If CHECKPOINT_COUNT is zero, the transfer is restarted from the beginning.
- If CHECKPOINT_COUNT is greater than zero, and a checkpoint has been reached, the transfer is restarted from the last confirmed checkpoint.
- If RESTART_SUPPORTED=NO, the transfer is not retried, regardless of the CHECKPOINT_COUNT value.

Default: YES

START_DATE

Indicates the date on which the daemon should start the scheduled transfer.

The format of START_DATE depends on the setting of the EURO_DATE parameter, as follows:

EURO_DATE value = YES

The format is DD/MM/YY.

EURO_DATE value = NO

The format is MM/DD/YY.

Default: None

Note: If START_DATE and START_TIME are not specified, the transfer starts immediately.

START_TIME

Indicates the time at which the daemon should start the scheduled transfer. The format of START_TIME is HH:MM:SS, in 24-hour military time.

Default: None

Note: If START_DATE and START_TIME are not specified, the transfer starts immediately.

Using the XCOMQM Program to Review Outstanding Transfers

To review all outstanding transfers, use the XCOMQM program to scan the checkpoint file. The XCOMQM program is similar to the CA XCOM Data Transport Queue Manager on UNIX or the Transfer Control menu option on z/OS. It lets you review and control scheduled transfers. It interfaces with the XCOMDMN process, discussed in the previous section.

Note: XCOMQM replaces the SCANCKPT program.

To start the XCOMQM program

Type the following command:

```
run vol.subvol.xcomqm vol.subvol.checkpoint_filename
```

The List option displays the requests in the CA XCOM Data Transport queue as shown below.

```
REQUEST    STAT    LOCAL ID  REMOTE ID       RECS     KBYTES  RETRIES  NEXT TIME
000040     S       USPRTU13  141.202.201.26  0        0       0        1997/08/30, 0:00:00
```

The following sections describe the various fields in the CA XCOM Data Transport queue.

You can list or purge records by request number, local ID, or status.

When scanning the checkpoint file, XCOMQM displays an entire screen of outstanding transfers that looks like this:

```
REQUEST      STAT      LOCAL ID   REMOTE ID    RECS        KBYTES   RETRIES NEXT TIME
000001        R        STRATT        LUICE        0        239        0    1999/12/05, 11:46:36
000002        R        STRATT        LUICE        0        239        0    1999/12/07, 10:47:08
No more CA-XCOM records
-- [M]ore, [Q]uit? [MQ]
```

You are then prompted with the options to scan more records. When no more records are found, you are prompted to do one of the following:

- [L]ist
- [P]urge
- [Q]uit the program

REQUEST

Request number assigned by CA XCOM Data Transport.

STAT

Status of the request.

The following list shows the meaning of the various status codes:

A

Active.

1

Active - Version 1 protocol.

S

Transfer is scheduled to be started at a future time.

R

Transfer can be retried.

I

Transfer is ineligible for restart.

D

The daemon program is in the process of preparing a transfer for CA XCOM Data Transport to start.

X

Remotely initiated transfer started, received CA XCOM Data Transport Header.

C

Process completion stage.

The number of remaining retry attempts and the next time the transfer is scheduled are also displayed.

LOCAL ID

ID of your system that originated the transfer.

REMOTE ID

The IP address of the remote logical unit.

RECS

The number of records transferred.

KBYTES

The number of kilobytes transferred.

RETRIES

The number of times the transfer has been tried.

NEXT TIME

The time scheduled for the next try.

NetBatch

NetBatch is a HP NonStop utility that allows you to set up queues for execution by a scheduler. This means that you can schedule CA XCOM Data Transport transfers at a specific time to a specific spooler location.

Note: Specify the TERM parameter in the NetBatch command as a spooler location.

You can schedule CA XCOM Data Transport transfers by creating TACL macros that include the CA XCOM Data Transport transfers. For examples of TACL command files containing CA XCOM Data Transport requests, see the chapter "The Batch/Command Line Interface." For more information about NetBatch, see your HP NonStop manuals.

Run CA XCOM Data Transport in the Background

To run a CA XCOM Data Transport process in the background

Use the following TACL command:

```
RUN XCOM62 /NOWAIT, TERM $S.#OUT/ command
```

NOWAIT

Indicates that CA XCOM Data Transport does not wait for a status response on the transfer.

TERM

Specifies the terminal to which CA XCOM Data Transport can have access.

If you specify your own terminal and you are not in PAUSE, CA XCOM Data Transport waits until it can access your terminal.

If you specify a spooler location (as in the example above), CA XCOM Data Transport writes all output to the spooler file. This allows you to review and print your results.

Note: If you do not specify a terminal, CA XCOM Data Transport uses your terminal for its process, and it hangs in the background until your terminal is available.

command

Specifies a standard CA XCOM Data Transport command.

Chapter 10: Security

For security, CA XCOM Data Transport for HP NonStop uses the following sets of user IDs and passwords:

- The Tandem Guardian logon
- The local CA XCOM Data Transport for HP NonStop user ID and password
- The remote CA XCOM Data Transport for HP NonStop user ID and password

Note: USERID and PASSWORD parameters are case-sensitive on some systems, so make sure that you use the correct case for the remote system.

This section contains the following topics:

HP NonStop Guardian Security Access

The person running CA XCOM Data Transport must have execute access to the program XCOM62, which in turn must have execute access to the APPC programs. The current account (the active Guardian logon) for CA XCOM Data Transport must have read access to the files EBCASC and ASCEBC (the files used for EBCDIC and ASCII translation).

HP NonStop Guardian Security Checking for Local Transfers

CA XCOM Data Transport enforces Guardian and SAFEGUARD file access rules. For locally initiated file transfers, CA XCOM Data Transport allows access to files according to the privileges of the HP NonStop user running CA XCOM Data Transport.

HP NonStop Guardian Security Checking for Remote Transfers

When CA XCOM Data Transport for HP NonStop receives a remote request, it verifies the HP NonStop user ID/password pair given by the remote user. This user ID/password pair could be a CA XCOM Data Transport pair or a Tandem pair. If it is a CA XCOM Data Transport pair, then the actual HP NonStop pair is retrieved from the password file. Guardian and SAFEGUARD file access control is used to determine if the HP NonStop user ID/password is valid and has access to the HP NonStop files.

Security Access

CA XCOM Data Transport for HP NonStop provides an encrypted password file for storing and retrieving records of user ID/password pairs. The default name for this password file is XCOMPWF. The remote CA XCOM Data Transport user ID/password pair is maintained by the PFILE2 program, which allows the creator of the record to assign security access. For information about the PFILE2 program, see Password File Maintenance in this chapter.

Note: Use of the password file during a CA XCOM Data Transport transfer is optional.

Security Checking for Local Transfers

A locally initiated transfer must pass the remote CA XCOM Data Transport user ID and password. This can be done with or without accessing the password file.

To use the password file

1. Set the USERID to a valid user ID for the partner being sent to. This same USERID must also be defined in the password file to be used.
2. Set the PASSWORD parameter to uppercase X. This value triggers the password file to be read.
3. Set the PASSWORD_FILE parameter to specify the password file name that was modified to add *both* of the following:
   - The USERID as specified in Step 1
   - The matching PASSWORD for this USERID

   Use the following format:

   vol.subvol.password_filename

   Note: The security access of the user ID/password pair record must match your HP NonStop logon.

To bypass the password file

Set the USERID and PASSWORD parameters to valid values for the remote CA XCOM Data Transport system being sent to.

Security Checking for Remote Transfers

A remotely initiated transfer must pass a user ID/password pair to the CA XCOM Data Transport process.

To use the password file

Set the PASSWORD_FILE parameter to the following password file name:

vol.subvol.password_filename

The remote CA XCOM Data Transport system passes a CA XCOM Data Transport user ID/password pair to the local CA XCOM Data Transport system. CA XCOM Data Transport for HP NonStop reads the CA XCOM Data Transport pair in the password file, and sends the matching HP NonStop logon to Guardian for security validation.

To bypass the password file, set the PASSWORD_FILE parameter to **NONE**. The password file is not checked. In this case, the remote CA XCOM Data Transport system must send a valid HP NonStop logon.

Password File Maintenance

The PFILE2 program lets you maintain your password file in the following ways:

- Add user ID/password pairs
- Delete user ID/password pairs
- Edit user ID/password pairs
- Display information about user ID/password pairs

To use PFILE2 to maintain your password file

1. Type the following command and press ENTER:

   ```
   run vol.subvol.PFILE2 vol.subvol.password_filename
   ```

   The program displays the following text:

   ```
   CA XCOM Data Transport PASSWORD MAINTENANCE FILE
   -- [A]dd, [D]elete, [E]dit, [L]ist, [Q]uit? [ADELQ]
   ```

2. Type the letter of the desired action and press ENTER.

   The system prompts you for the following security information:

   Important! Data is case sensitive.

   ```
   Enter CA XCOM Data Transport User ID:
   Enter CA XCOM Data Transport Password:
   Re-enter CA XCOM Data Transport Password:
   Enter TANDEM User: (e.g.: SUPER.SUPER)
   Enter TANDEM Password:
   Re-enter TANDEM Password:
   Security Access? [U]ser only, [G]roup, [A]ll:
   ```

Local CA XCOM Data Transport User ID/Password Pairs

If you are adding a local CA XCOM Data Transport user ID/password pair, you need a matching HP NonStop logon, but the security access specified for that pair is ignored.

Remote CA XCOM Data Transport User ID/Password Pairs

If you are adding a remote CA XCOM Data Transport user ID/password pair, ignore the HP NonStop logon prompts. The security access determines whether other CA XCOM Data Transport users are authorized to use this pair.

Information About User ID/Password Pairs

The LIST option of PFILE2 displays the following information about the selected user ID/password pair:

- User
- Date/time stamp of the last access
- Whether or not there is a password associated with the user ID

Chapter 11: Generating SSL Certificates

This chapter describes how to generate certificates that can be used with CA XCOM Data Transport. For more information about using OpenSSL, see *Network Security with OpenSSL* by John Vega, Matt Messier, and Pravir Chandra (O'Reilly & Associates).

This section contains the following topics:

Using SSL Mode

CA XCOM Data Transport uses SSL in client/server mode. In client/server mode, certificates are required for both the local (initiating) and remote (receiving) CA XCOM Data Transport partners. SSL considers the local CA XCOM Data Transport partner to be the client and the remote CA XCOM Data Transport partner to be the server.

When establishing the SSL connection, the server sends the server certificate to the client for verification. After the client verifies the server certificate, the client sends the client certificate to the server for verification. Both the client and the server must verify the CA certificate from the other.

Important! Certificates should be created by super.super.

Setting up SSL for CA XCOM Data Transport involves the following tasks:

1. Create the CA certificate.
2. Set the expiration for the CA certificate.
3. Create the server certificate.
4. Create the client certificate.
5. Configure the CA XCOM Data Transport SSL server.
6. Configure the CA XCOM Data Transport client.

Set Expiration

When generating a CA certificate, the default_days parameter in cassl.conf that controls the expiration of server and client certificates is not used for CA certificates. The certificate is generated with a default expiration of 365 days.

To change the default expiration

1. Add 'days *nnn*' to the makeca script line. The following line is an example of how the makeca script is shipped:

   ```
   Openssl req –x509 –newkey rsa:2048 -config caconf –out casslpem –outform PEM
   ```

2. To change the expiration to 30 days, change the line as follows before running the makeca script:

   ```
   Openssl req –x509 –newkey rsa:2048 -config caconf –out casslpem –outform PEM –days 30
   ```

Create the CA Certificate

To create the CA certificate

1. Create a configuration file that is used as input to the openssl utility. A sample file, named caconf, was installed in the volume.subvolume of the CA XCOM Data Transport installation directory for HP NonStop. Change the HP NonStop and HP NonStop Integrity volume.subvolume and edit the [root_ca_distinguished_name] section, changing the values as appropriate for your system.
2. Issue the following command to run the makeca script:

   ```
   makeca
   ```

   This TACL macro uses the caconf to generate a certificate and key file. The certificate, casslpem, and the key file, generated as cakeypem, are saved in the installation subvolume.

   Note: When running the makeca script the first time, the pseudo-random number generator (PRNG) file does not exist and issues a warning to this effect. The makeca utility generates the PRNG file the first time it is run and does not issue this warning on subsequent executions. This is only a warning; you can continue with the next step.

3. To list the certificate just created, issue the following command to use the listca TACL macro:

   ```
   listca
   ```

   This TACL macro displays the CA certificate and the information stored in the package.

Create the Server Certificate

To create the server certificate

1. Create a configuration file to use as input to the openssl utility. A sample file, srvconf, was installed in the volume.subvolume of the CA XCOM Data Transport installation directory for HP NonStop and HP NonStop Integrity. Edit the [req_distinguished_name] section, changing the values to your specifications.
2. Using the TACL macro makesrv, issue the following command:

   ```
   makesrv
   ```

   The makesrv TACL macro uses the srvconf file and the casslpem file to generate a server certificate and a key file. The server certificate, srvcpem, and the key file, generated as srvkpem, are saved in the installation subvolume.
3. To list the certificate just created, issue the following command to use the listserver script:

   ```
   listsrv
   ```

   This TACL macro displays the server certificate and information stored in the package.

Create the Client Certificate

To create the client certificate

1. Create a configuration file to use as input to the openssl utility. A sample file, cltconf, was installed in the installation volume.subvolume. Edit the [req_distinguished_name] section, changing the values to meet your system requirements.
2. Issue the following command to use the makeclt TACL macro:

   ```
   makeclt
   ```

   The makeclt TACL macro uses the cltconf file and the casslpem file to generate a client certificate and a key file. The certificate, cltcpem, and the key file, generated as cltkpem, are saved in the installation volume.subvolume.
3. To list the certificate just created, issue the following command to use the listclt TACL macro:

   ```
   listclt
   ```

   The listclt TACL macro displays the client certificate and information stored in the package.

Configure the CA XCOM Data Transport SSL Server

To configure CA XCOM Data Transport to use the CA and server certificates for establishing server (remote) SSL connections

1. Review and modify the CA XCOM Data Transport SSL configuration file, configssl.cnf (or, for HP NonStop, xcsslcnf), so that the settings meet your site standards. Server connections use the RECEIVE_SIDE values.
2. Set the XCOM_CONFIG_SSL parameter in your default options table/global file to point to your customized configssl.cnf file.

   Note: For z/OS, the path and file name must be an HFS file.
3. Configure CA XCOM Data Transport to receive remote SSL connections, as follows:
   - For z/OS, specify the TCP/IP port that will accept SSL connection requests using the SSLPORT default options table parameter. In addition, the default options table parameter, SSL, must also be set to one of the following values:
     - ONLY—to allow incoming SSL transfers only
     - ALLOW—to allow both incoming SSL and incoming non-SSL transfers to this server
   - For UNIX, during installation, manually add the txpis service and the TCP/IP port that will accept SSL connection requests to the inetd configuration files.
   - For HP NonStop, during installation, manually add the XCOM62 program and the TCP/IP port that will accept SSL connection requests to the PORTCONF configuration files.
   - For Windows, specify the TCP/IP port that that will accept SSL connection requests using the SSL Port Number on the TCP/IP tab in the Global Parameters GUI.
4. Verify that the port that receives incoming SSL connections is a unique port that is not in use by any other application. The port used for incoming TCP/IP connections cannot also be used for incoming SSL connections. If CA XCOM Data Transport will be receiving both incoming TCP/IP connections and incoming SSL connections, then two ports are required.
   - For z/OS, reassemble the default options table and restart the CA XCOM Data Transport server (started task).
   - For UNIX and Windows, restart the CA XCOM Data Transport service.

Configure the CA XCOM Data Transport Client

To configure the CA XCOM Data Transport client to use the CA certificate and the server certificate when establishing client (local) SSL connections

1. Review and modify the settings of the CA XCOM Data Transport SSL configuration file, configssl.cnf (or, for HP NonStop, xcsslcnf), as appropriate for your system. Client connections use the INITIATE_SIDE values.
2. Point the XCOM_CONFIG_SSL parameter in your default options table/global file to your customized configssl.cnf file.

   Note: For z/OS, the path and file name must be an HFS file.

   - For z/OS, the XCOM_CONFIG_SSL parameter can also be specified as a destination member parameter.
   - For HP NonStop, UNIX, and Windows, the XCOM_CONFIG_SSL parameter can also be specified in your configuration (cnf) file.
3. Set the SECURE_SOCKET parameter to YES to indicate an SSL connection.
   - For z/OS, specify the SECURE_SOCKET parameter in the SYSIN01, the destination member, or the default options table.
   - For UNIX and Windows, specify the SECURE_SOCKET parameter in the configuration (cnf) file or in the global (xcom.glb) file.
   - For HP NonStop, specify the SECURE_SOCKET parameter in the XCOMCNF global file, in a user-defined configuration (xcomcnf) file, or on the command line.
4. Specify the port through which the remote CA XCOM Data Transport partner accepts SSL connections. Use one of the following parameters:
   - PORT for HP NonStop, UNIX, and Windows
   - IPPORT for z/OS
5. Initiate the transfer request.

Example of Generating SSL Certificates

When testing with SSL certificates, you need to create and/or use a ROOT CA when generating certificates on your systems.

Note: Certificate Authority (CA) Administration can be administered in a number of ways. If you are currently using certificates, you may want to check with your Certificate Authority Administrator.

The following example will use the casslpem and cakeypem generated on SysA as your ROOT CAs for SysA and SysB.

1. On SysA, run all of the following:
    a. MAKECA
    b. MAKESRV
    c. MAKECLT
2. On SysB, remove all existing certs/pems, index, and serial files that may be present:
    a. purge idx
    b. purge srl
    c. purge randpem
    d. purge casslpem
    e. purge cakeypem
    f. purge rand
    g. purge idxanew
    h. purge idxnew
    i. purge idxold
    j. purge idxa
    k. purge idxaold
    l. purge srlnew
    m. purge srlold
    n. purge srvcpem
    o. purge srvkpem
    p. purge srvrqpem
    q. purge x01pem
    r. purge cltkpem
    s. purge cltcpem
    t. purge cltrqpem

3. On SysB, run MAKECA.
4. On SysB, purge the casslpem and cakeypem files created by the MAKECA:
   a. purge casslpem
   b. purge cakeypem
5. Copy the casslpem and cakeypem created on SysA to SysB.

   Note: If the SysA ROOT certificates were generated on an EBCDIC-based system such as z/OS or i5/OS, you need to convert the certificates from EBCDIC to ASCII.

   Example:

   If the ROOT certificates were created from CA XCOM Data Transport for z/OS or CA XCOM Data Transport for AS/400, you would need to copy both of the following:

   - certs/cassl.pem to CASSLPEM, converting EBCDIC to ASCII
   - private/casslkey.pem to CAKEYPEM, converting EBCDIC to ASCII
6. Verify, using the listca utility, that the root certificate is correct.
7. On SysB, run the following:
   a. a. MAKESRV
   b. b. MAKECLT
8. Verify, using the listsrv and listclt utilities, that the server and client certificates are correct.

Chapter 12: Remote System Information

This chapter contains information about important aspects of the operating systems supported by CA XCOM Data Transport that you should be aware of when performing transfers.

For more specific information about operating CA XCOM Data Transport on a specific platform, see the CA XCOM Data Transport guides for that platform and the manufacturer's guides.

The following topics are covered for each platform, as appropriate:

- Naming conventions
- Types of files supported
- Additional features
- Restrictions

HP NonStop (Tandem)

This section contains information about important aspects of the HP NonStop operating system.

Naming Conventions—HP NonStop (Tandem)

The following list describes the parts of an HP NonStop file name:

system

Specifies the system name. Up to seven characters.

volume

Specifies the disk name.

subvolume

Specifies a directory name.

filename

Specifies the name of your file.

Example:

The following example uses a volume of $CLX12, a subvolume of SCI, and a file name of FILE1:

```
$CLX12.SCI.FILE1
```

The HP NonStop file system is not a tree structure. Each volume.subvolume is independent, that is, it has no subvolumes above or below.

Types of Files Supported—HP NonStop (Tandem)

CA XCOM Data Transport for HP NonStop supports the following file types through ENSCRIBE, Tandem's disk file architecture:

- Edit files
- Unstructured files

  Unstructured files are large-byte arrays. Data in these files is accessed by using the relative byte address and the READ-COUNT or WRITE-COUNT parameters in the system procedure calls. The application program determines the way in which they are used. An EDIT file is a type of unstructured file signified by the file code 101.

  For more information about ENSCRIBE and unstructured files, see the *ENSCRIBE Programmer's Guide*.
- Structured files

  CA XCOM Data Transport supports entry-sequenced, relative, and key-sequenced structured files:

  - Entry-sequenced files

    Entry-sequenced files are sequential files. Records are stored in the order in which they are entered. These records are variable in length and cannot be added or deleted. They are accessed by their record address.
  - Relative files

    Relative files are ordered by relative record number. The space allocated for each record is specified when the file is created. Records in these files can be deleted and added again in place.
  - Key-sequenced files

Key-sequenced files are supported only for the Replace operation. The file must already exist for CA XCOM Data Transport to perform an action on it.

File Type Specification—HP NonStop (Tandem)

File type specification differs for send requests and received requests, described as follows:

- Send Requests

  When you send a file from HP NonStop (locally initiated), the remote CA XCOM Data Transport determines the file type when it opens the file.

- Receive Requests

  For locally or remotely initiated receive requests, the file type must be specified by the GUARDIAN_FILE_TYPE parameter. Use one of the following values:

  - EDIT
  - UNSTRUCTURED
  - ENTRYSEQ
  - RELATIVE

Remotely Initiated Send Requests—HP NonStop (Tandem)

For remotely initiated transfer requests (for example, send a file, job, or report), use the following record formats, as shown in the following table, which create the indicated guardian file types:

Record Format	Description
F	Relative
FB	Entry Sequence
VB	Edit
U	Unstructured

Note: Key sequence files are supported only if the file exists. You can do a replace but not a create.

i5/OS (AS/400)

This section contains information about important aspects of the i5/OS operating system.

Naming Conventions—i5/OS (AS/400)

Use the following format to specify an i5/OS file:

libraryname/filename(membername)

The following list describes the parts of an i5/OS file name:

libraryname

The name of the library that holds the file.

filename

The name of the file you wish to access. Periods are allowed within the file name.

membername

The name of the member in the file. If this component is omitted, it defaults to the file name.

Types of Files Supported—i5/OS (AS/400)

In addition to the standard file type discussed above, the Save File format is also supported. When you wish to send such a file to a System i5 from a z/OS or z/VSE system, the file must exist on the target system prior to your transmission.

Additional Features—i5/OS (AS/400)

XQUE is a CA XCOM Data Transport feature that allows the unattended transfer of reports from output queues to other CA XCOM Data Transport nodes.

XQUE can select specific classes of reports (based on the user, job name, form, and so on) from output queues. XQUE also allows user and workstation groups to be equated to printer destinations on remote CA XCOM Data Transport nodes. You can use XQUE, for example, to get reports back to your host system that are generated on a System i5 that you reach through IBM's HCF facility, or between multiple i5/OS (AS/400) systems connected within a pass-through environment.

Configuration Issues—i5/OS (AS/400)

If you are configuring the VTAM LU that represents the System i5 on a mainframe, make sure that the VTAM USS message 10 is not sent to that LU. IBM's APPC software cannot start a session when this message, commonly called the welcome message, is sent.

To prevent this problem, the VTAM or NCP USSTAB definition must be set to a table that does not have a USSMSG10. The table that IBM originally provided with VTAM is a good alternative because it does not include message 10.

Case Sensitivity—i5/OS (AS/400)

Because the IBM i5/OS is case-sensitive, you must enter the user ID and password in uppercase.

Novell NetWare

This section contains information about important aspects of the Novell NetWare operating system.

Naming Conventions—Novell NetWare

Use the following format to name a Netware file:

Note: CA XCOM Data Transport for LAN Workstation accesses files from any Novell file server in a NetWare network.

`[server\]volume:directory\subdirectory\...\filename`

Types of Files Supported—Novell NetWare

CA XCOM Data Transport for NetWare LAN supports standard NetWare file types.

Destination Printer Information—Novell NetWare

When sending a report to a NetWare system, specify the Destination parameter value or the Destination Printer field in the following form:

\\server name\printer queue name

CA XCOM Data Transport limits the length of this field as indicated below. The actual name on the destination system can be longer.

Direct transfers using Version 2 protocols

Specify up to 21 characters.

Indirect transfers or transfers using Version 1 protocols

Specify up to 16 characters.

Restriction—Novell NetWare

CA XCOM Data Transport for NetWare LAN does not support library transfers to Novell NetWare systems.

OpenVMS

This section contains information about important aspects of the OpenVMS operating system.

Naming Conventions—OpenVMS

Use the following format to name an OpenVMS Alpha file:

device[*directory*]*filename.type*;*version*

The entire file specification can be a maximum of 255 characters. The file type can be a maximum of 31 characters.

The following list describes the parts of an OpenVMS file name:

device

Specifies the disk drive name. If the device is not specified, the default provided in the SYSUAF (as defined on the DEC system) for that user is used. Range: 1 to 15 characters.

Note: The CA XCOM Data Transport remote USERID field determines the SYSUAF USERID.

directory

Specifies the directory and subdirectory information. If this information is not provided, defaults are selected as described under “device” above.

Note: CA XCOM Data Transport accepts angle brackets (< >) in OpenVMS file names, which are converted to square brackets on the DEC system.

Example:

PLAYERS1:Unicenter Management for eTrust Security Command CenterCARD.DAT is treated as equivalent to
PLAYERS1:[BRIDGE]CARD.DAT

filename.type

Specifies the specific file within the directory. OpenVMS null file names are used if the file name and type are not provided.

version

Specifies the version of the file. The OpenVMS operating system can keep multiple versions of a file each time that file is saved. It is normal to omit this number to indicate that you want the most recent version of a file, the highest version number.

For more information about OpenVMS file specifications, see the OpenVMS documentation.

Restrictions—OpenVMS

The following restrictions apply to CA XCOM Data Transport for OpenVMS Alpha:

- Specifying transfer type

  All transfers must be TYPE=SCHEDULE (for batch) or QUEUED (from ISPF).

- Non-queued host transfers

  Due to restrictions in the DEC SNA software, the z/OS or z/VSE TYPE=EXECUTE (non-queued) transfer feature fails with an 8003 sense code. It is not supported by CA XCOM Data Transport to an OpenVMS system.

- Operating system

  CA XCOM Data Transport currently supports the OpenVMS Alpha operating system.

- Connectivity

  Specifies the DECNET/SNA software is based on the Physical Unit 2.0 standard and not on the more flexible 2.1. This means that the system must be connected to a VTAM (PU 5) system in an SNA network. CA XCOM Data Transport uses the store-and-forward function (described previously as an additional z/OS, z/VM, and z/VSE feature) to transfer files with other CA XCOM Data Transport partners.

- Multiple session configuration

Stratus VOS

This section contains information about important aspects of the Stratus operating system.

Naming Conventions—Stratus VOS

Use the following format to name Stratus files:

#top_directory>group_directory>home_directory>filename.suffix

All names must be unique to that level.

Important! CA Technologies recommends that you use only UNC conventions for all mapped or redirected drives while sending data to a Windows system.

The following list describes the parts of a Stratus file name:

top_directory

Specifies the physical disks.

Range: 1 to 32 characters.

group_directory

Specifies a group of user home directories.

Range: 1 to 32 characters.

home_directory

Specifies the user's home directory. This directory resides in a group directory.

Range: 1 to 32 characters.

filename

Specifies the name of the Stratus file.

Required.

Range: 1 to 32 characters.

suffix

Specifies a file classification. You can have multiple suffixes at the end of a file name. Each suffix starts with a period. The following list describes some common Stratus suffixes:

source

.pl1, .cobol, .c

Examples:

payroll.c, application.cobol

object

.obj

Examples:

payroll.obj, application.obj

list

.list

Examples:

payroll.list, application.list

error

.error

Examples:

payroll.error, application.error

program module

.pm

Examples:

payroll.pm, application.pm

command macro

.cm

Examples:

start_up.cm, compile_and_bind.cm

back up

.backup

Examples:

payroll.c.backup

Types of Files Supported—Stratus VOS

Stratus supports the following file types for remotely initiated transfers:

- Fixed

  This type of file contains records of the same size. Each record is stored in a disk or tape region holding a number of bytes that is the same for all the records in the file.

- Sequential

  This type of file contains records of varying sizes in a disk or tape region holding approximately the same number of bytes as the record (for example, the record storage regions vary from record to record). Records can only be accessed on a record-by-record basis.

Additional Features—Stratus VOS

The following are additional features of CA XCOM Data Transport for Stratus of which you should be aware:

- Security option

  CA XCOM Data Transport for Stratus can use its own account file to verify the user ID and password and to map the CA XCOM Data Transport user ID to a VOS user ID to check for file access. If this option is turned on and the remote user ID/password combination is invalid, CA XCOM Data Transport for Stratus rejects the request.

- Restart/Recovery facility

  CA XCOM Data Transport for Stratus can attempt periodic data transmissions after the initial file transfer has failed. A certain number of retries can be specified through the xcom_ser.pm file.

Restrictions—Stratus VOS

The following restrictions apply to CA XCOM Data Transport for Stratus:

- No checkpoint/restart

  CA XCOM Data Transport for Stratus Version 1 does not support checkpoint/restart.

- No library transfers

  CA XCOM Data Transport for Stratus does not support the transfer of libraries from the mainframe.

UNIX or Linux

This section contains information about important aspects of the UNIX or Linux operating systems.

Naming Conventions—UNIX or Linux

Use the following format to name a UNIX or Linux file:

/directory/subdirectory/.../filename

Use up to 256 characters for the entire path of the file; there are no restrictions on size for the individual parts of the path.

The following list describes the parts of a UNIX or Linux path:

/ (slash)

The root directory when it is in the first position: otherwise, the slash separates directories and file names in the path.

directory

Specifies the directory that contains the file. You can specify more than one directory in a path.

filename

Specifies the name of the UNIX or Linux file.

Types of Files Supported—UNIX or Linux

CA XCOM Data Transport for UNIX or Linux supports standard UNIX or Linux file types.

Restriction—UNIX or Linux

CA XCOM Data Transport does not support library transfers to UNIX or Linux systems.

Windows

This section contains information about important aspects of the Windows operating systems.

Naming Conventions—Windows

CA XCOM Data Transport supports the standard Windows file names and the Universal Naming Convention (UNC). Some of the file naming conventions are outlined below.

Use the following format to name files when using standard Windows file names:

d:[\][*directory name*\..\]*filename*[.*ext*]

Important! This format may only be used if the drive is a local drive on the Windows system. Do not use for mapped or redirected drives. Use UNC conventions only for mapped or redirected drives.

Note: Use the following format to name files when using UNC file names:

```
\\server name\share name\directory\filename.
```

The following list describes the parts of the file names and UNC file names:

d

Required. Specifies a particular device, indicated as a drive letter. Used for local drives only.

directory name

Required. One or more optional directories and subdirectories.

Subdirectories can take the form of *name[.ext].*

Note: The form of the directory name and file name depend on the operating system running on the server.

filename

Required. Specifies the name of the data file.

For FAT file systems, *filename* is 1 to 8 characters.

NTFS and HPFS file systems support long file names, up to 256 characters, including the extension.

Names may or may not be case sensitive, depending on the file system on the server.

For FAT, NTFS and HPFS, names are not case sensitive. You can use uppercase and lowercase when creating a name, and they display as typed, but internally Windows makes no distinction for this. For example, MYFILE and MyFiLe are considered to be the same file.

Windows also creates an MS-DOS-style name based on the long name for compatibility with environments where long file names are not always supported.

ext

The file extension used to further identify the file.

For FAT file systems, the extension is up to 3 characters.

For NTFS and HPFS, the extension is included in the long file name limit of 256 characters.

Note: If you do not specify an extension, CA XCOM Data Transport does not supply a default.

server name

The name of the server.

share name

The share name is network provider dependent.

For Microsoft Windows networks this is the name of the share.

Types of Files Supported—Windows

CA XCOM Data Transport supports standard Windows file types.

Destination Printer Information—Windows

When sending a report to a Windows system, specify the Destination parameter value or the Destination Printer field in the following form:

`\\server name\printer queue name`

CA XCOM Data Transport limits the length of this field as indicated below. The actual name on the destination system can be longer.

Direct transfers using Version 2 protocols

Specify up to 21 characters.

Restrictions—Windows

Access to directories and files on drives formatted for NTFS can be controlled with the security features of Windows 2000, XP, or 2003.

Access to all files on a Windows system can be controlled by the permissions set on a directory or file. The access rights of the user ID on the remote system determine the actions permitted for the transfer. Users cannot use a directory or file unless they have been granted the appropriate permissions.

z/OS

This section contains information about important aspects of the z/OS operating system.

Naming Conventions—z/OS

Use the following format to name a z/OS file (data set):

[*level1.level2.level3...level7*].*level8*[(*membername*)]

The following table describes the parts of a z/OS file name:

level

Specifies the level of a file name. Required.

A file name can consist of multiple levels separated by a period. Each level has the following characteristics:

- It can be up to eight uppercase characters long.
- It starts with either an alphabetic character or a national character ($, #, @, +, -, :, _).

There is a limit of eight levels with a total of 44 characters, including the separating periods.

In most z/OS environments, a data set name is further restricted by security rules created by the installation. Contact the appropriate personnel within your organization for details. Typically, the high-level name (first-level name) must match your z/OS user ID or some other predefined index.

membername

Specifies the particular member in a z/OS partitioned data set (PDS). A PDS is a library containing members that are each separate sequential files. The member name is appended to the end of the file name in parentheses.

Required for z/OS partitioned data sets only.

Range: One to eight alphanumeric or national characters.

Note: Most sites catalog all files through the system master catalog. In short, this means that the system can locate the file you specify by name only. With the rare occurrence of an un-cataloged file, you need to specify the volume and unit information for the device that holds the file.

Example:

The following are examples of valid z/OS data set names:

```
SYS1.VTAMLST
C54684.UTILITY.CNTL(JOBCARD)
PROD.PAYROLL.SEPT90.TIMECARD.DATA
TESTDATA
A.$DDD.LOAD
```

Types of Files Supported—z/OS

Sequential files are the most common forms of data transferred. Individual members of PDS files can also be sent as sequential files. Entire PDS libraries or multiple selected members can be transferred between two z/OS systems or to other systems running CA XCOM Data Transport r11 or higher. PDSE and entire PDSE program libraries are supported in CA XCOM Data Transport starting at r11. PDSE program libraries do not support wildcarding.

All three types of VSAM files (KSDS, ESDS, and RRDS) can be transferred between z/OS systems. These VSAM files must be pre-allocated, or they can be sent to non-z/OS systems as sequential files.

UNIX System Services (USS) files are also supported where an entire file system is stored in a single z/OS data set.

ISAM, BDAM (direct access), IMS, FDR, and DFDSS data sets are not directly supported, but they can be put into a sequential format using native utilities prior to transmission.

DCB Information—z/OS

The file characteristics for z/OS must be predefined when creating a new file. Collectively, the following characteristics are known as Data Control Block (DCB) parameters:

- Block size
- Logical record length
- Record format
- Volume
- Unit

For more information regarding any of these fields, see the IBM JCL reference manual.

z/VM

This section contains information about important aspects of the z/VM operating system.

Naming Conventions—z/VM

Use the following format to name z/VM files under the CMS operating system:

filename.filetype

The two parts can be a maximum of eight characters in length. They can consist of letters, numbers, and/or national characters ($, #, @, +, -, :, _). In general, lowercase letters are not allowed. In the CA XCOM Data Transport for z/VM parameters FILE and LFILE, the file name and file type are specified as one string with a period as a separator.

For minidisk specifications:

- CP OWNER is taken from the volume field, if present. Otherwise, the userid field is used.
- CP address is taken from the unit specification. The default is 191.
- You can have two files with the same file name and file type, but they cannot reside on the same minidisk.

Types of Files Supported—z/VM

The CA XCOM Data Transport Service Virtual Machine runs IBM's Group Control System (GCS) operating system. Due to the limitations of this environment, CA XCOM Data Transport for z/VM only supports the CMS extended file system format. This covers CMS files on minidisks formatted with 512 KB, 1,024 KB, 2,048 KB, and 4,096 KB block sizes.

Note: It does *not* support the following: CMS Shared File System, minidisks formatted with 800-byte blocks, or tape I/O.

DCB Information—z/VM

CMS file characteristics must be predetermined when creating a new file. You must specify the following parameters:

- Record format

  This can be fixed (F) or variable (V).

- Logical record length

  This is the number of characters in the longest line of the file.

Restriction—z/VM

The maximum logical record lengths for different file types are as follows:

- **Disk file**

  32767 bytes

- **Job (RDR file)**

  80 bytes

- **Report (PRT file)**

  133 bytes

z/VSE

This section contains information about important aspects of the z/VSE operating system.

VSAM Naming Conventions—z/VSE

When accessing a file on a z/VSE system, the Remote file name field indicates the file ID as it would be specified on the DLBL (an indicator of whether the file is VSAM or SAM) and, optionally, additional information needed for locating the file.

Format for VSAM File Names

Use the following format to name a VSAM file:

file-id,V[,*catalog-id*]

The following list describes the parts of a VSAM file name:

file-id

Specifies the name given to the data set when it was defined using IDCAMS by including the following line in the JCL:

```
DEFINE CLUSTER (NAME   (file-id)...
```

V

Indicates that this is a VSAM file.

catalog-id

Optional.

The name of the user catalog that owns the VSAM data set as defined using IDCAMS by including the following line in the JCL:

```
DEFINE USERCATALOG (NAME (catalog-id)...
```

Leave this field blank if the data set is owned by the master catalog.

Format for SAM File Names

Use the following format to name a SAM file:

file-id,S,[*unit*],[*location*],[*size*],[*override*]

The following list describes the parts of a SAM file name:

file-id

The name that identifies this data set in the VTOC of the specific DASD volume. This is the file ID you specify on the DLBL JCL statement. Range: 1 to 44 characters.

Note: Do not enclose it in quotes.

S

Indicates that this is a SAM file.

unit

The physical device address as defined by the CUU parameter on the ASSGN JCL statement. It identifies the disk drive on which this file resides. This parameter can be omitted if the UNIT or VOL parameters are specified, or if a DASD manager is in use.

location

Optional for output files.

The starting location of the file on the disk, as defined on the EXTENT JCL statement. If a DASD manager is in use, specify a value of 1.

size

Optional for output files.

Indicates how much space this data set is to use, as defined on the EXTENT JCL statement. For CKD devices, this is the number of tracks. For FBA devices, this is the number of blocks.

override

Optional for output files.

The following override parameters can be specified. The equivalent DFLTAB option is included to describe each override parameter:

- DMYES to force DASDM=YES for this file
- DMNO to force DASDM=NO for this file
- DMEPIC to force DASDM=EPIC for this file

Note: If you are running with a DASD manager, the DASD manager's STRTTRK or Trigger value would be placed in the location field. DASD manager pools should be indicated by putting the pool name in the Volume parameter.

For EPIC/VSE users, you can omit the following:

- The location if you want EPIC to default to its STRTTRK value.
- The size if you want EPIC to default to its DEFEXT value.
- The Volume information if you want EPIC to default to its DEFPOL value.

For CA Dynam/T users who want to access Dynam catalog controlled files (included GDG data sets), no extent information should be entered. (No *cuu*, location, size, or override information and no Volume or Unit parameters for the files you are referencing.)

TAPE Naming Conventions

Use the following format to name a TAPE file:

file-id,T,[*unit*],[*unit*],[*unit*],[*override*]

The following list describes the parts of a TAPE file name:

file-id

Specifies the name that identifies this data set in the tape manager catalog or in the HDR1 label on the tape. This is the file ID you specify on the TLBL JCL statement. Range: 1 to 44 characters.

Note: When the file ID contains imbedded spaces or commas, it should be enclosed in quotes.

Note: IBM only supports a 17-character file ID in a tape header label. If you have a tape manager, 44-character tape file IDs can be supported. CA XCOM Data Transport does not validate your file ID, but takes whatever you put on the statement and passes it to IBM's OPEN routine or to your tape manager as you have entered it.

T

Indicates that this is a TAPE file.

Note: If you enter a transfer request from a platform that has not yet implemented the extended tape processing parameters or menu interface fields for controlling tape processing to a CA XCOM Data Transport z/VSE server, you must use the T option. You are restricted to standard label tape processing.

unit

The physical device address as defined by the CUU parameter on the ASSGN JCL statement. If you are using TAPEM=YES|EPIC, CA XCOM Data Transport ignores any units coded and the tape manager does the tape AVR and assignment. If you are not using the tape manager, the primary assignment is made to the first unit CA XCOM Data Transport finds. Other units found are assigned as temporary alternates.

This parameter can be omitted if you prefer to use the UNIT parameter to specify a unit or two units (primary and alternate). This parameter can be used in conjunction with the UNIT parameter to specify a primary unit and up to four alternate units that are to be assigned by CA XCOM Data Transport prior to open. Units specified on the statement containing the file ID are assigned before units specified on the UNIT parameter. The unit parameter is ignored because tape processing is only supported when you have a tape manager on your z/VSE system.

override

Optional for output files.

The following override parameters can be specified. The equivalent DFLTAB option is included to describe each override parameter:

- TMYES to force TAPEM=YES for this file
- TMNO to force TAPEM=NO for this file
- TMEPIC to force TAPEM=EPIC for this file

Note: The override applies only to the processing for the file whose data set name is on the statement that the override appears on. It is in effect for this transfer only.

VSAM Managed SAM Naming Conventions

Use the following format to name a VSAM managed SAM file:

file-id,M,*prim#recs*, *sec#recs*,*catalog-id*

The following list describes the parts of a VSAM managed SAM file name:

file-id

The name that identifies this data set, which is implicitly defined to VSAM at open time.

Range: 1 to 44 characters.

M

Indicates that this is a VSAM managed SAM file.

Note: If you enter a transfer request from a platform that has not yet implemented the extended tape processing parameters or menu interface fields for controlling tape processing to a CA XCOM Data Transport z/VSE server, you must use the T option. You are restricted to standard label tape processing.

prim#recs

Used for output files only. This indicates the number of blocks (of the size defined by the BLKSIZE parameter) for the primary data set allocation.

sec#recs

Used for output files only. This indicates the number of blocks for the secondary data set allocation. If no secondary allocation is coded, VSAM defaults to 20% of the primary allocation. Zero can be specified if you do not want any secondary allocation.

catalog-id

Optional for output files.

Defines the name of the user catalog that will own the data set. You can leave this field blank if the master catalog owns the data set.

Note: The use of VSAM managed SAM files requires IBM's IDCAMS program to be dynamically loaded in the partition. This requires an additional 130 KB partition GETVIS storage.

DTF Information

z/VSE file characteristics must be predetermined when creating the files. If sending to or receiving from a z/VSE system you must specify the following:

- The record format (RECFM), which can be either fixed (F), fixed blocked (FB), variable (V), or variable blocked (VB).
- The logical record length (LRECL) indicates the number of characters in the longest record in the file.
- The block size (BLKSIZE), which must be one of the following:
  - The LRECL for fixed files
  - A multiple of the LRECL for fixed blocked files
  - The LRECL +4 for variable files
  - The BLKSIZE +4 for variable blocked files

Types of Files Supported—z/VSE

IBM z/VSE supports VSAM (RRDS, KSDS, and ESDS) and SAM files.

Restrictions—z/VSE

The following restrictions apply to CA XCOM Data Transport for z/VSE:

- No FILEOPT=ADD for receiving z/VSE

  CA XCOM Data Transport for z/VSE does not support FILEOPT=ADD if the z/VSE is receiving the file.

- No Checkpoint/Restart for SAM

  CA XCOM Data Transport for z/VSE does not support checkpoint/restart for SAM jobs.

Appendix A: Configuration File Parameters

This appendix contains an alphabetical listing of the parameters in the default configuration file.

This section contains the following topics:

List of Parameters

ALLOC_UNIT

Used only when creating mainframe files.

Specifies the size of the allocation unit if the remote is an IBM mainframe. The actual byte count of each type will vary, depending on the storage device.

B

Blocks

C

Cylinders

T

Tracks

Default: B

Note: If you have questions about allocation units, consult your System Administrator.

APPC_PROCESS_NAME

The name of the process used by CA XCOM Data Transport. This name must agree with the process name specified in the SNAX/APC configuration.

Example:

If you used the supplied PATHCOLD file to start SNAX/APC, the APPC_PROCESS_NAME is $SNAS.

Range: Up to 16 characters

Default: None

APPC_TYPE

Required.

Indicates your APPC configuration type.

Range: SNAXAPPC or TCPIP

Default: SNAXAPPC

ASCEBC

Specifies which file to use for ASCII to EBCDIC conversion.

If a file name is entered, CA XCOM Data Transport uses that file for the translation. If there is no value entered, or if CA XCOM Data Transport cannot find the file, CA XCOM Data Transport uses the default settings (the same as the deliverable tables).

The parameter format is as follows:

ASCEBC=*vol.subvol.filename*

Note: If you enter commands from different subvolumes, you must specify the full *vol.subvol.filename* for this parameter. Make sure that all users of this file have the correct access.

Range:

- Up to 8 characters for *filename*
- Up to 26 characters for *vol.subvol.filename*

Defaults:

- For *filename*: ascebc
- For *vol.subvol.filename*: None

BLKSIZE

Specifies the physical block size of a file. The range depends on record length.

For a variable record format

BLKSIZE = LRECL + 4

For a fixed or fixed blocked record format

BLKSIZE = a multiple of LRECL

For an undefined record format

BLKSIZE > largest record length

Note: If you create a structured file on the HP NonStop system, it must be a valid HP NonStop block size. CA XCOM Data Transport computes an appropriate value.

Range: Up to five characters

Default: 4096

CACHEBUF

Writes records to cache instead of directly to disk. If the cache buffer becomes full, the records are written to disk.

Cache buffering is a standard Guardian option. Because cache buffering is set on each disk drive, performance varies from disk to disk. For more information, see the PUP manual.

YES

Turns cache buffering on.

NO

No cache buffering.

Default: NO

CARRIAGE_CONTROL_CHARACTERS

Indicates the type of carriage control characters that are used in the print job.

ASA carriage control characters are as follows:

Blank

Space 1 line

0

Space 2 lines

-

Space 3 lines

+

Suppress space

1

Skip to line 1 on new page

Valid options are as follows:

ASA

ASA control codes in column 1.

- When sending a disk file from Tandem to print on a remote system, specify ASA if the disk file has ASA carriage control characters in column one. Otherwise, specify OTHER. You should choose IBM only if you previously sent a file from a mainframe with machine carriage control characters to a Tandem disk file, and now want to send it back to a mainframe for printing.
- When the XQUE feature is specified, this parameter controls whether ASA codes are generated when CA XCOM Data Transport sends the file from the Tandem spooler to the remote system.
- When a report is sent to a Tandem system, the Tandem interprets the ASA characters if the remote partner specified ASA. Tandem does not support the IBM machine code carriage control characters.

IBM

IBM Machine Characters (valid for a z/OS remote system only).

OTHER

No carriage control codes.

Default: OTHER

CARRIAGE_FLAG

Controls the treatment of text files

If CARRIAGE_FLAG=YES and CODE_FLAG is ASCII or EBCDIC, new line characters are added to incoming records and removed from outgoing records.

The Tandem file system, like mainframe and AS/400 file systems, does not use record separators. When transferring text files with a system that does use record separators (UNIX or PC), make sure that the other system adds them when receiving files and removes them when sending files.

Range: YES or NO

Default: YES

CHARS

Reports only.

Specifies the font for reports sent to a z/OS system. For more information, see your z/OS manual.

Range: Up to four characters

Default: None

CHECKPOINT_COUNT

Specifies the number of records between checkpoints.

Note: This parameter is not recognized unless VERSION=2.

Range: 0000 to 9999

Default: 0000

CHECKPOINT_FILE

Specifies the name of the checkpoint file to which the checkpoint requests are written.

Use the following format:

vol.subvol.filename

Because of changes in the layout of the transfer record to accommodate TCP/IP, when upgrading from a previous version the checkpoint file must be redefined, as follows:

```
FUP PURGE CKPTFIL
FUP PURGE CKPTALT
FUP /IN MKCKPT/
```

Range: Up to 27 characters

Default: ckptfil

CLASS

Reports only.

Indicates the print class for the print job.

If the remote system is a z/OS system, then CLASS designates the JES SYSOUT class. In this case, to print the report through SYSOUT=B, enter B.

Note: If printing on HP NonStop, this parameter is ignored.

Range: One character

Default: None

CODE_FLAG

Identifies the type of data being transferred.

Important! CA XCOM Data Transport translates every byte in the file. If you have mixed character and binary data, the file will be corrupted if you specify EBCDIC.

EBCDIC

Translation is required when sending a file.

ASCII

Translation is required when receiving a file.

BINARY

No translation is required. Specify BINARY if a binary file such as an executable file is being transferred.

Default: ASCII

CODETABL

Applies to Windows, Linux, and UNIX partners only.

Specifies the prefix to the custom character conversion file names on Windows, Linux, or UNIX that will be used by the transfer.

Range: Zero to three alphanumeric characters

Default: None

COMPRESS

Indicates the transmission type. Compressing data may decrease transmission time.

NO

Do not compress the data transmission buffers. This option is used when the CPU resource is more of a constraint than network bandwidth.

YES

The original CA XCOM Data Transport compression method for reducing strings of multiple blanks and nulls. Provided for backward compatibility.

RLE

Run length encoding of any repeating characters. This is the least CPU intensive of the compression methods.

COMPACT

Run length encoding of any repeating characters, plus a two-byte compaction algorithm suitable for uppercase English text.

LCOMPACT

Run length encoding of any repeating characters, plus a two-byte compaction algorithm suitable for mixed case English text.

LZSMALL | LZMEDIUM | LZLARGE

Lempel-Ziv derivatives for small, medium, and large memory models. They achieve the greatest reduction in the data transmitted, but consume the most CPU time.

HUFFMAN

This option selects a basic Huffman encoding technique. This technique tends to provide greater compression than RLE, but not as much as the Lempel-Ziv 77 derivatives. Huffman uses more CPU than RLE, but generally uses less than the Lempel-Ziv 77 derivatives.

LZRW3

The LZRW3 algorithm is a general purpose compression algorithm that runs quickly and gives reasonable compression. The algorithm is a member of the Lempel-Ziv family of algorithms, and bases its compression on the presence of repeated substrings in the data.

Next to RLE, this is the least expensive compression option in terms of CPU utilization. It will not reduce the data transmitted by as much as ZLIB, LZSMALL, LZMEDIUM, and LZLARGE, but it usually consumes far less CPU time. With some data, this method will use less CPU time than HUFFMAN, while providing greater compression.

ZLIB*n*

Greater compression than LZRW3 but less than LZSMALL, LZMEDIUM, and LZLARGE. The *n* value can be 1 through 9. ZLIB is a Lempel-Ziv 77 derivative. This technique tends to provide greater compression than LZRW3, but somewhat less than LZSMALL, LZMEDIUM, and LZLARGE. CPU utilization tends to be much greater than LZRW3 and RLE, but somewhat less than LZSMALL, LZMEDIUM, and LZLARGE.

Default: YES

Compression usage guidelines are as follows:

- When sending text files to an IBM AS/400, COMPRESS should be set to YES to overcome the problem of zero-length lines. Compression guarantees that all lines will have at least one character to satisfy the LU 6.2 read on the receiving end.
- Most of the current CA XCOM Data Transport releases support all compressions. Check the product documentation for each platform for details.
- COMPRESS=YES is provided for backward compatibility with older releases of CA XCOM Data Transport. For any current release, COMPRESS=RLE is a better choice.
- COMPRESS=RLE is inexpensive in terms of CPU utilization and, for text files, is recommended over COMPRESS=NONE.
- COMPRESS=LZRW3 is the least expensive of the advanced compression methods in terms of CPU utilization, and should be tried first. If you are CPU bound, you may get better wall clock time using RLE, COMPACT, COMPACTL, or LZRW3, rather than HUFFMAN, ZLIB, LZSMALL, LZMEDIUM, or LZLARGE.
- COMPACT and COMPACTL add a byte compaction scheme on top of RLE. They may compress text files slightly better than RLE, without adding much in terms of CPU utilization.
- Use Huffman, or any of the Lempel-Ziv 77 derivatives, only if you are using packing (for example, PACK=BIG) or have an LRECL of 500 or greater. The output buffer that CA XCOM Data Transport tries to compress must be at least 500 bytes long for these compression methods to be effective. In some cases compression is disabled if the output buffer is less than 500 bytes. If you do not use packing and your LRECL is less than 500 bytes, then you should use RLE, COMPACT, or LCOMPACT.

  For maximum benefit from Huffman or any of the Lempel-Ziv 77 derivatives, you should code PACK=BIG and IO_BUFFSIZE=32000 in the XCOMCNF file.
- The slower the communications link, the more important compression is to data transfer speeds. Conversely, the faster the communications link, the less important compression will be. If you have a fast communications link, you may see little difference in wall clock time between transfers using COMPRESS=RLE versus transfers using COMPRESS=LZLARGE.

- When examining CPU utilization, you have to compare both the time it takes to compress the data and the time it takes to expand the data. Particularly with the Lempel-Ziv derivatives, compression tends to take more CPU time than decompression. This consideration could be important if one of the transfer partners is more CPU constrained than the other.
- The choice of which compression method to use depends on several factors:
  - Communications line speed
  - CPU utilization constraints
  - Nature of the repetitiveness of the data

For your routine production jobs, you are encouraged to experiment with the various compression methods to see which one will provide the best compromise between network I/O and CPU utilization for your data in your environment.

COMPRESS_PDS

Applies to z/OS only.

COMPRESS_PDS is the parameter that causes the actual PDS compression to happen. If your CA XCOM Data Transport z/OS administrator has enabled the programmatic PDS compression feature in a CA XCOM Data Transport region, you can use the COMPRESS_PDS option to control if and when output PDS data sets get compressed as part of the transfer.

Note: COMPRESS_PDS applies only to PDS data sets that will be, or have been, opened for output as the target of a CA XCOM Data Transport transfer.

NONE

Suppresses the compression of an output PDS data set as part of a CA XCOM Data Transport transfer.

BEFORE

Causes an output PDS data set to be compressed before the transfer of user data begins.

AFTER

Causes an output PDS data set to be compressed after the transfer of user data has completed.

BOTH

Causes an output PDS data set to be compressed both before and after the transfer of user data.

Default: NONE

CONV_SECURITY

Applies to locally initiated SNAX transfers.

Specifies whether the user ID/password pair is to be sent in the SNA ATTACH request. On the mainframe, CONV_SECURITY is controlled by the ACCSEC parameter in the CA XCOM Data Transport Destination Table (XCOMCNTL).

YES

Sends the user ID/password pair in the ATTACH request.

NO

User ID/password pair is not sent in the ATTACH request.

Default: NO

COPIES

Reports only.

Indicates the number of copies to be printed when a remote system sends a report to CA XCOM Data Transport for HP NonStop.

Range: Up to three characters

Default: 1

CREATEDELETE

Applies to z/OS only.

CREATEDELETE specifies whether an existing z/OS data set should be deleted and a new data set allocated at the start of a FILE_OPTION=CREATE transfer.

YES

If FILE_OPTION=CREATE and the data set exists, then the z/OS data set is deleted and a new data set is allocated at the start of the transfer.

NO

If FILE_OPTION=CREATE and the z/OS data set exists, then the transfer fails with a catalog/file error.

Default: NO

Notes:

- Specifying CREATEDELETE=YES causes the attributes of the existing data set to be lost; the new data set is allocated with the attributes specified in the transfer.
- CREATEDELETE applies only if the target data set is a sequential data set or an entire PDS/PDSE. CREATEDELETE is ignored for other types of data sets (such as PDS members, PDSE members, VSAM, and USS files).
- CREATEDELETE does not apply to relative GDGs unless the data set is specified using the fully qualified G*xxxx*V*xx* name.
- The use of CREATEDELETE=YES must be allowed by your site's CA XCOM Data Transport administrator for z/OS through the default table (XCOMDFLT) or destination member (XCOMCNTL).

DATACLAS

Specifies the name of the data class to use when allocating a new SMS-managed data set.

Note: This parameter applies only to mainframe SMS data sets.

Range: One to eight characters

Default: None

DEALLOC_EXTENTS

Returns any unused extents to the system when a file is closed.

If set to YES, a CONTROL 21.0 is executed to deallocate all unused extents past the end of file.

Range: YES or NO

Default: NO

DEN

Specifies the density to be used in creating a tape on the remote system. Valid values are the same as those for the DEN parameter in JCL.

Range: 1 to 4

Default: None

DESTINATION

Reports only.

Indicates the print job's destination on the remote system. If no destination is specified, the remote system sends the job to the system's default printer.

For report printing on Tandem systems, the remote system should specify the destination as follows:

`$<collector>.#<location>`

If no COLLECTOR is specified, then SPOOL_COLLECTOR is used.

Range: Up to 21 characters

Default: None

DIR_ALLOC

Specifies the number of directory blocks to allocate when creating a PDS data set on a remote z/OS system. This corresponds to MAXEXTENTS on HP NonStop.

Range: 0 to 32767

Default: 0

DISPOSITION

Reports only.

Indicates the disposition of the printed file after printing is completed. Whether this parameter is meaningful is system dependent.

DELETE

Delete the file after it is printed.

KEEP

Do not delete the file.

HOLD

Hold after printing.

Default: DELETE

DOMAIN

The Windows domain name for use in authenticating the user ID and password when accessing a Windows based machine that has sharable disks and drives that belong to that domain. This allows users to access these sharable drives without having to have a local user ID or password defined to the machine.

Range: 1 to 15 characters

Default: None

DSNTYPE

Specifies the data set definition.

Note: This parameter applies only to mainframe SMS data sets.

LIBRARY

Defines a PDSE.

PDS

Defines a partitioned data set.

Note: These values are IBM standards for SMS processing.

Range: One to eight characters

Default: None

EBCASC

Specifies the file to use for EBCDIC to ASCII conversion.

If a file name is entered, then CA XCOM Data Transport uses that file for the translation. If there is no value entered, or if CA XCOM Data Transport cannot find the file, CA XCOM Data Transport uses the default settings (the same as the deliverable tables).

The parameter format is as follows:

EBCASC=*vol.subvol.filename*

Note: If you enter commands from different subvolumes, you must specify the full *vol.subvol.filename* for this parameter. Make sure that all of the users of this file have the correct access.

Range:

- Up to 8 characters for *filename*
- Up to 26 characters for *vol.subvol.filename*

Defaults:

- For *filename*: ebcasc
- For *vol.subvol.filename*: None

EURO_DATE

The EURO_DATE parameter determines the format of START_DATE, as follows:

EURO_DATE value = YES

The format of START_DATE is DD/MM/YY.

EURO_DATE value = NO

The format of START_DATE is MM/DD/YY.

Default: NO

EXPDT

Specifies an expiration date for the tape data set in terms of a two-digit designation for the year and a three-digit designation for the day of the year.

Example:

In the expiration date 11021, 11 is the year (namely, 2011) and 021 is the 21st day of that year, when the tape data set expires.

Format: *yyddd*

Default: None

Note: EXPDT and RETPD are mutually exclusive; specify one or the other.

FCB

Indicates the forms control block (FCB) JCL parameter when sending the report file to a z/OS mainframe. It defines print density, lines per page, and so on.

Note: FCB is ignored for report printing on HP NonStop systems.

Range: Up to four characters

Default: None

FILE_CODE

Specifies the Guardian Enscribe file code when creating a file on the local HP NonStop system.

Range: 0 to 9999

Default: 0

FILE_OPTION

Indicates how the transferred data is to be processed by the receiving system.

CREATE

Creates a new file on the receiving system.

APPEND

Appends this data to an existing file on the receiving system.

REPLACE

Replaces the contents of an existing file on the receiving system. On HP NonStop, if the file does not exist, it is created automatically.

Default: CREATE

FORM

Specifies which forms the printed output should use.

When a remote system sends a report to CA XCOM Data Transport for HP NonStop, the FORM parameter must identify a valid Tandem Spooler form.

Valid names can contain letters, digits, and blanks only. Invalid characters result in the transfer being failed with a SPOOLSTART error 4097.

Because CA XCOM Data Transport places the print job in the remote system's print queue, the print control functions will depend on the remote system. Before sending the report, you must verify that the form you are requesting is available at the remote site.

Note: When sending a report to an OpenVMS system, leave FORM blank unless you are certain that the value is a valid form type. OpenVMS interprets a blank to mean that no special form is being requested.

Range: Up to 10 characters

Default: The default form for the remote printer

GATEWAYGUID

Identifies the remote file as a CA XCOM Gateway file and specifies the CA XCOM Gateway GUID. This is a unique value that identifies each CA XCOM Gateway file.

Note: When the CA XCOM Gateway GUID is not known, the keyword ANY can be used to identify the remote file as a CA XCOM Gateway file.

Range: 0 to 36 characters

Default: None (the remote file is not a CA XCOM Gateway file)

GUARDIAN_FILE_TYPE

HP NonStop Disk File Creation parameter.

Indicates the type of Enscribe file to create.

- For a locally initiated transfer, specify either EDIT, RELATIVE, ENTRYSEQ, or UNSTRUCTURED to create that type of disk file.
- For a remotely initiated transfer, or if GUARDIAN_FILE_TYPE=NONE or is not specified, the value of RECORD_FORMAT determines the type of disk file to create.

For more information, see Handling HP NonStop File in the chapter "Configuring CA XCOM Data Transport."

Range: EDIT, RELATIVE, ENTRYSEQ, UNSTRUCTURED, or NONE

Default: NONE

HISTORY_FILE

Specifies the name of the history file to which the history records are written. Use the following format:

vol.subvol.filename

Range: Up to 27 characters

Default: xcomhist

HOLD_FLAG

Reports only.

Indicates the transferred report file's HOLD status on the remote system.

Valid options are as follows:

YES

Hold the report (spooled on a z/OS system).

NO

Prepare the report for immediate printing.

Default: NO

IPC_FNAME

Specifies the program name (and optional startup parameters specific to your program) that will run if the process specified in the IPC_PNAME does not exist.

The file name is specified in external format. Use a space to delimit the filename from the startup parameters, and the startup parameters from each other, as follows:

```
$CLX01.EXAMPLE.MYAPPL  PARM1 PARM2 PARM3
```

Note: If the full path name is not specified, the XDIR parameter supplies the missing volume and subvolume names.

Range: 1 to 150 characters, beginning with the character $

Default: None

IPC_NO_REMOTE

Specifies if IPC information received from a remote system is ignored.

YES

Any IPC information provided by a remote system is ignored.

NO

IPC information provided by a remote system is not ignored.

Default: NO

IPC_PNAME

The process name from which CA XCOM Data Transport reads data or to which it sends data, entered in the following format:

```
<$process><.#qualifier1><.#qualifier2>
```

If an IPC_FNAME is not specified, the IPC process must be running already.

If the IPC process is not running, CA XCOM Data Transport starts one from the IPC_FNAME information.

Note: An IPC_PNAME is required for locally initiated transfers.

Range: 1 to 24 characters, beginning with the character $

Default: None

IO_BUFFSIZE

Used with SNAX/APC.

IO_BUFFSIZE lets you maximize throughput and eliminate excessive overhead in interprocess communication between CA XCOM Data Transport and SNAX/APC. This parameter is used to specify the size of the buffer when the Tandem sends a file to another system.

Notes:

- When Big Packing is used, this parameter controls how large the pack buffers will be (similar to the MAXPACK parameter in the CA XCOM Data Transport for z/OS Destination Table (XCOMCNTL)).
- When Big Packing is not being used, this parameter controls the size of the buffers passed from the CA XCOM Data Transport process to the SNAX process.
- For SNAX, the IO_BUFFSIZE value must be less than or equal to the SNAX/APC MAXAPPLIOSIZE parameter value. In general, the IO_BUFFSIZE should be higher for higher speed lines.

Range: From 4136 to 32000, inclusive

Default: 31744

JOB_TIME_OUT

Specifies the period of time that CA XCOM Data Transport is to wait for a send job to complete. You can instruct CA XCOM Data Transport to wait no longer than *nnnnn* seconds for a send job to complete.

Zero means no waiting.

Range: 0 to 86400

Default: 0

Note: If a remote send job takes longer than the number of seconds specified for JOB_TIME_OUT, the transfer terminates with an error 40 (the operation timed out).

With such an error 40, even if the parameter DISPOSITION is set to DELETE, the temporary file holding the remote TACL commands TEMP*xxxx* does not get deleted and the job is left running. It is up to the user to investigate, stop the job, purge the TEMP*xxxx* file, and take corrective action (increase the JOB_TIME_OUT value or change the processing) so that the error 40 does not occur again.

LABELNUM

Indicates the sequence number of the data set on the tape.

Sequence number (0001 to 9999)

This value identifies the sequence number of a data set on tape.

Example:

LABELNUM=2

This specification refers to the second data set on the tape.

Default: 0001

LCLNTFYL

Specifies the local user notification level.

ALL

NOTIFY on transfer completion.

WARN

NOTIFY only if the transfer received a warning or error.

ERROR

NOTIFY only if the transfer received an error.

Default: ALL

LOCAL_FILE

Identifies the name of the file on the local system. Tandem file naming conventions apply.

Important! For scheduled transfers, you must specify the full path name.

Range: Up to 256 characters

Default: None

LOCAL_NOTIFY

Specifies which user to notify on the local system when CA XCOM Data Transport has completed the transfer.

Range: Up to 64 characters

Default: None

LRECL

Specifies the actual or maximum length in bytes of a logical record. This corresponds to the JCL LRECL subparameter.

For a variable blocked format

LRECL should equal the maximum record length.

For a fixed or fixed blocked format

LRECL should equal the constant record length.

Range: Up to five characters

Default: 0, except in the following cases:

- If GUARDIAN_FILE_TYPE=EDIT or UNSTRUCTURED, the default is 239, or 243 for variable blocked.
- If GUARDIAN_FILE_TYPE=RELATIVE or ENTRYSEQ, the default is taken from the record length parameter in the transferred file.

MAXEXTENTS

Sets a limit lower than the default for this file type:

- For EDIT files, the default is 900.
- For all other file types, the default is 256.

Notes:

- For a remotely initiated transfer, the DIR_ALLOC parameter is used.
- For a z/OS initiated transfer, this is the last item in the SPACE parameter.

MGMTCLAS

Specifies the name of the management class to use when allocating a new SMS-managed data set.

Note: This parameter applies only to mainframe SMS data sets.

Range: One to eight characters

Default: None

NOTIFY_NAME

Specifies which user to notify on the remote system when CA XCOM Data Transport has completed its procedure.

If the remote system is a z/OS system, CA XCOM Data Transport uses the value of NOTIFYR to determine the type of notification to deliver.

If the remote system is an HP NonStop system, the user receives a mail message.

Range: Up to 12 characters

Default: None

NOTIFYR

Specifies the notification flag on the remote system.

TSO

TSO user notification.

WTO

Write to log only.

CICS

CICS user notification.

LU

Logical unit notification.

VM

VM/CMS user notification.

NONE

No user notification.

Note: This parameter is associated with the NOTIFY_NAME parameter.

Default: NONE

NULLFILL

Indicates whether CA XCOM Data Transport for HP NonStop is to fill the end of outgoing EBCDIC text records with null characters.

N

Do not use null characters at the end of the record.

Y

Use null characters at the end of the record.

Default: NO

PACK

Packs up to 31KB of data into a buffer before transmission to a remote system. The receiving system is responsible for unpacking the record(s).

NO

Does not use record packing. Each logical record is sent out individually.

YES

Uses packing feature with a 2KB buffer.

Notes:

- On Windows, Linux, and UNIX platforms, this is specified by CARRIAGE_FLAG=MPACK.
- On z/OS, this is specified by the combination of PACK=LENGTH and MAXPACK=2048.

BIG

Uses packing feature with up to a 31KB buffer.

Notes:

- On Windows, Linux, and UNIX platforms, this is specified by CARRIAGE_FLAG=XPACK.
- On z/OS, this is specified by the combination of PACK=LENGTH and MAXPACK with a value greater than 2048.
- When HP NonStop is sending the file, the packing block size is determined by the IO_BUFFSIZE parameter.
- When HP NonStop is receiving the file, the remote partner determines the packing block size.

Note: The mainframe versions of CA XCOM Data Transport support a version of packing where records are separated by line end characters. On z/OS, this is specified by PACK=CRLF. The HP NonStop version of CA XCOM Data Transport does not support this record packing method. When you define an HP NonStop partner in the CA XCOM Data Transport for z/OS Destination Member (XCOMCNTL), you should code PACK=LENGTH and MAXPACK=31744.

Default: NO

PASSWORD

Indicates the remote password to use with the file security scheme on the remote system.

Range: Up to 31 characters

Default: None

PASSWORD_FILE

Specifies the name of the CA XCOM Data Transport security file. Use the following format:

vol.subvol.filename

Range: Up to 27 characters or NONE

NONE

Disables the CA XCOM Data Transport security feature.

Note: Setting PASSWORD_FILE=NONE disables CA XCOM Data Transport security only. It does not affect Tandem security. For more information, see the chapter "Security."

Default: NONE

PORT

The number of the TCP/IP port on the remote CA XCOM Data Transport server. Used for TCP/IP transfers only.

Range: 1 to 65535

Default: 8044

PRI_ALLOC

The primary extent size for creating local and remote files.

Range: 1 to 32767

Default: 2

RECORD_FORMAT

Specifies the record format for the file being created. This corresponds to the JCL RECFM subparameter.

F (Fixed Unblocked)

All records have the same length.

FB (Fixed Blocked)

Fixed record length with multiple records per block.

VB (Variable Blocked)

Variable record length with multiple records per block.

U (Undefined)

Undefined record length.

Default: VB

REMOTE_FILE

Indicates the name of the file on the remote system.

If you are creating the file, make sure your designated file name is consistent with the file naming conventions of the remote system. The remote system (not the local CA XCOM Data Transport system) determines whether the file name is valid.

Range: Up to 256 characters

Default: None

REMOTE_SYSTEM

For SNA

The LU name

For TCP/IP

The remote system's IP address, host name, or domain name

For indirect transfers

(That is, for store-and-forward transfers to CA XCOM Data Transport for z/OS or z/VSE that have another final destination), the REMOTE_SYSTEM name is the name of the final destination as defined in the CA XCOM Data Transport destination table on the mainframe.

Range: Up to 128 characters

Default: XCOMAPPL

REPORT_TITLE

Reports only.

Provides the report name to be printed on the job separator when a remote system sends a report to CA XCOM Data Transport for HP NonStop.

Valid names can contain letters, digits, and blanks only. Invalid characters result in the transfer being failed with a SPOOLSTART error 4097.

The title is interpreted depending on the type of remote (receiving) system, as follows:

i5/OS (AS/400)

CPF assumes this to be the printer file name.

z/OS

A non-blank value generates a separator (banner) page.

OpenVMS

This title will be printed with the report.

UNIX

This field will be passed to the lp spooler as a title field.

Other systems

This field is generally used only as a descriptive comment and is not printed as part of the report.

Range: Up to 21 characters

Default: None

REQUEST_NO

System-generated.

Specifies the unique ID number associated with each transfer. Used for restart requests.

Range: Up to six characters

Default: None

RETPD

Specifies the number of days (1 to 9999) that the tape data set being created is to be retained.

Range: 1 to 9999

Default: None

Note: RETPD and EXPDT are mutually exclusive; specify one or the other.

RETRIES

Indicates the number of times a transfer should be retried by the daemon.

Note: A transfer cannot be resumed or restarted unless this parameter is set to a value greater than zero.

Range: 0 to 32767

Default: 0

RETRY_TIME

Indicates number of seconds the daemon program should wait for a transfer to be started.

Range: 0 to 32767

Default: 60 seconds

RECYCLE (HP NonStop Parameter)

Indicates number of seconds the daemon program should wait between scans of the checkpoint file.

Note: Be aware of the settings of both the RECYCLE and RETRY_TIME parameters. For example, if the RETRY-TIME is set to 60 seconds, but the RECYCLE parameter is set to 300 seconds, then the transfer will not restart for at least 60 seconds. However, it may be 300 seconds until the daemon program actually reads the checkpoint file and restarts the transfer.

Default: 300 seconds (five minutes)

REMOTE_EXPIRE (HP NonStop Parameter)

Number of seconds the daemon program should wait before purging remotely initiated transfers marked for restart (status R displayed by XCOMQM).

Default: 300 seconds (five minutes)

RESTART_FLAG

Indicates if a transfer is a restart request. This parameter can be used to force a restart.

Range: Y or N

Default: N

RESTART_SUPPORTED

Determines whether a locally initiated transfer can be started, as follows:

- If CHECKPOINT_COUNT is zero, the transfer is restarted from the beginning.
- If CHECKPOINT_COUNT is greater than zero, and a checkpoint has been reached, the transfer is restarted from the last confirmed checkpoint.
- If RESTART_SUPPORTED=NO, the transfer is not retried, regardless of the CHECKPOINT_COUNT value.

Default: YES

RLOGFILE

Specifies the name and location of the log file for remotely initiated transfers. You may specify as little of the path name as you like.

Example:

Suppose CA XCOM Data Transport creates the remote log file name as follows:

```
$SYSTEM.CAXCOM.RL971101
```

Then you could specify the following RLOGFILE values to create the corresponding file names:

$DSV

$DSV.CAXCOM.RL971101

$DSV.XCOM

$DSV.XCOM.RL971101

$DSV.XCOM.RLOG

$DSV.XCOM.RLOG

If you specify only volume or volume/subvolume, a new file name is created when the day changes.

If RLOGFILE is not specified, the value specified for XDIR is used. If neither RLOGFILE nor XDIR is specified, the default volume used is $SYSTEM.

The default subvolume is as follows:

- CAXCOM for TCP/IP transfers
- The LU name for SNA transfers

If RLOGFILE is set to NONE, no file is created and no logging information is written.

RLOG_SECURITY

When using SNAX in a remotely initiating scenario, CA XCOM Data Transport acts as a server handling multiple, serial transfer requests from the SNAX Dispatcher. This makes the system more efficient since it is not necessary to create a new CA XCOM Data Transport process for every transfer. For each transfer, CA XCOM Data Transport logs in as the user requested in the transfer.

Remote log and trace files are named by the system date (for example, RT950418). When a new file is created during the server's existence, it uses the user ID of the latest successful transfer, and thus this file inherits the ownership and security attributes of that user ID. It is not possible for CA XCOM Data Transport to revert back to the original user ID used at process creation time without the password.

If you don't want the new log file to be created with the last transfer's user ID, set the new parameter RLOG_SECURITY to Y. If there is a need to create a new log file, CA XCOM Data Transport will stop, the SNAX Dispatcher will create a new CA XCOM Data Transport process, and the trace and log files will be created with the correct user ID.

Range: Y or N

Default: N

RMTNTFYL

Specifies the remote user notification level when sending data to a remote system.

ALL

NOTIFY on transfer completion.

WARN

NOTIFY only if the transfer received a warning or error.

ERROR

NOTIFY only if the transfer received an error.

Default: ALL

RTRACEFILE

Specifies the name and location of the trace file for remotely initiated transfers. You may specify as little of the pathname as you like.

Example:

Suppose CA XCOM Data Transport creates the remote trace file name as follows:

```
$SYSTEM.CAXCOM.RT971101
```

Then you could specify the following RTRACEFILE values to create the corresponding file names:

$DSV

$DSV.CAXCOM.RT971101

$DSV.XCOM

$DSV.XCOM.RT971101

$DSV.XCOM.RTRACE

$DSV.XCOM.RTRACE

If you specify only volume or volume/subvolume, a new file name is created when the day changes.

If RTRACEFILE is not specified, the value specified for XDIR is used. If neither RTRACEFILE nor XDIR is specified, the default volume used is $SYSTEM. The default subvolume is as follows:

- CAXCOM for TCP/IP transfers
- The LU name for SNA transfers

If RTRACEFILE is set to NONE, no file will be created and no trace information will be written.

SEC_ALLOC

The secondary extent size for creating local and remote files.

Range: 1 to 3567

Default: 4

SECURE_SOCKET

Specifies whether to use an OpenSSL socket or non-OpenSSL socket for transfers.

YES

Performs a secure transfer.

The transfer uses an OpenSSL socket and must to connect to a SSL listener on the remote partner.

NO

Performs a non-secure transfer.

The transfer uses a non-OpenSSL socket.

Default: NO

SOCK_DELAY

The default TCP/IP Socket option is TCP_NODELAY. This parameter refers to the Nagle algorithm for send coalescing. By default, small sends may be delayed, but this should have no impact for normal CA XCOM Data Transport record sizes. Used for TCP/IP transfers only.

Note: Socket options affect the way CA XCOM Data Transport uses TCP/IP stack implementation.

YES

Small sends may be delayed. (Does not turn on the socket option TCP_NODELAY, and does not disable the Nagle algorithm)

NO

All sends are immediate. (Disables the Nagle algorithm)

Default: YES

SOCK_RCV_BUF_SIZE

The default TCP/IP Socket option is SO_RCVBUF. This parameter can be used to specify the buffer size for receives. Use zero for the default size provided by the socket implementation. The value for SOCK_RCV_BUF_SIZE can be smaller than the value for TXPI_BUF_SIZE. Used for TCP/IP transfers only.

Note: Socket options affect the way CA XCOM Data Transport uses the TCP/IP stack implementation.

Range: 0 to 32760

Default: 0

SOCK_SEND_BUF_SIZE

The default TCP/IP Socket option is SO_SNDBUF. This parameter can be used to specify the buffer size for sends. Use zero for the default size provided by the socket implementation. The value for SOCK_SEND_BUF_SIZE can be smaller than the value for TXPI_BUF_SIZE. Used for TCP/IP transfers only.

Note: Socket options affect the way CA XCOM Data Transport uses the TCP/IP stack implementation.

Range: 0 to 32760

Default: 0

SPOOL_COLLECTOR

Indicates the default location for reports received from a remote system. For jobs received from a remote system, the output of the job will be written to this spool collector.

The format is as follows:

$<collector>.#<location>

collector

The name of a spooler process. The standard Tandem default spooler collector name is $S.

location

Similar to a job name on other systems.

When CA XCOM Data Transport for HP NonStop receives a job file, a new TACL process is created to execute the file. The output file for this job is the file defined for SPOOL_COLLECTOR.

Range: Up to 25 characters

Default: $S.#XCOMJOB

SPOOL_FLAG

System-dependent flag.

Indicates to the remote system whether it should spool the report received. HP NonStop sends all the reports that it receives to the spooler.

YES

Spool the report received from the local system.

NO

Do not spool the report.

Default: YES

SPOOL_JOBNUMBER

When a remote system performs a SEND JOB to HP NonStop, this parameter controls the value of the jobid parameter passed to the Guardian NEWPROCESS procedure call.

YES

The job ID of the CA XCOM Data Transport process is passed to NEWPROCESS. The CA XCOM Data Transport process is the ancestor of the newly created TACL process. If the number of jobs for each spooler queue is limited by the Tandem administrator, and you reach the limit, then the job fails with error 14.

NO

No job ID is supplied to the NEWPROCESS call. If the CA XCOM Data Transport process is not part of a job, neither is the new process. If the CA XCOM Data Transport process is part of a job, the new process is part of the same job.

ZERO

A value of zero is passed to the NEWPROCESS call. The new process will not be part of any job.

The number of jobs for each spooler queue can be limited by the HP NonStop administrator.

- If SPOOL_JOBNUMBER=YES, CA XCOM Data Transport honors this limit.
- If SPOOL_JOBNUMBER=NO, CA XCOM Data Transport ignores this limit.

Range: YES, NO, and ZERO

Default: YES

START_DATE

Indicates the date on which the daemon should start the scheduled transfer.

The format of START_DATE depends on the setting of the EURO_DATE parameter, as follows:

EURO_DATE value = YES

The format is DD/MM/YY.

EURO_DATE value = NO

The format is MM/DD/YY.

Default: None

Note: If START_DATE and START_TIME are not specified, the transfer starts immediately.

START_TIME

Indicates the time at which the daemon should start the scheduled transfer. The format of START_TIME is HH:MM:SS, in 24-hour military time.

Default: None

Note: If START_DATE and START_TIME are not specified, the transfer starts immediately.

STORCLAS

Specifies the name of the storage class for a new SMS-managed data set.

Note: This parameter applies only to mainframe SMS data sets.

Range: One to eight characters

Default: None

SYSTEM_USER_DATA

Specifies system-dependent user data included with each transfer. Only used with Version 2 partners.

Range: Up to 10 characters

Default: None

TAPE

Indicates to the remote system whether the volume is a tape volume or a disk file.

YES

Indicates a tape volume and that mounts are allowed when performing dynamic allocation.

NO

Indicates that the transfer is to a disk file.

Default: None

TAPE_LABEL

Indicates the type of label associated with a tape data set. The following table lists the valid values for this parameter.

Processing type (AL, AUL, BLP, LTM, NL, NSL, SL, SUL)

Represents the type of processing to be applied to data sets on tape.

Note: CA XCOM Data Transport for z/OS supports only standard label tapes.

Example:

LABEL=BLP

The type of processing to be applied to this data set is BLP.

Default: AL

TAPEDISP

Specifies the disposition value for MVS tape data sets.

1

New

2

Old

3

Mod

Default: 1

TCP_RECEIVE_TIMEOUT

Specifies the period of time that CA XCOM Data Transport will wait for a TCP socket call to complete.

You can instruct CA XCOM Data Transport to wait no longer than *nn* seconds for a TCP socket call to complete. On HP NonStop, CA XCOM Data Transport uses nowait TCP socket calls, followed by the AWAITIOX system call. This parameter is used by AWAITIOX to determine how long to wait for completion before assuming failure. Zero means an unlimited wait.

Range: 0 to 999

Default: 60 (one minute)

TRANSFER_ID

Specifies the non-unique user-assigned identifier for each transfer.

Range: Up to 10 characters

Default: None

TRANSFER_USER_DATA

Indicates any user-specified information for each transfer that can be passed to the remote system. This information is written in the history record.

Range: Up to 10 characters

Default: None

TRUSTED

Allows the user to request a trusted transfer and the partner's CA XCOM Data Transport TRUSTED database to be searched to verify the user's credentials. This eliminates the need for the user to specify a USERID and PASSWORD. If XCOM_TRUSTED_OVR is set to NO or no USERID is specified, the USERID of the process that initiated the transfer will be used.

Note: TRUSTED=YES cannot be specified with indirect transfers, because this is not supported.

Range: YES, NO, Y, N

Default: NO

TXPI_BUF_SIZE

Specifies the internal buffer size for sends and receives for TCP/IP transfers. The default size allows multiple CA XCOM Data Transport records to be received in a single socket call. With this default, CA XCOM Data Transport will attempt to receive multiple records in a single socket call, if the CA XCOM Data Transport record size is less than 32K. Used for TCP/IP transfers only.

Range: 0 to 65536

Default: 32760

TXPI_SEND_CHECK_FREQ

Indicates how frequently CA XCOM Data Transport checks to see if incoming error information is available when sending data.

For example, if the value is 5, a check is made every fifth time that data is sent, to determine if data is available for receiving. Larger values give better performance. Smaller values minimize the sending of data after the partner reports an error. Used for TCP/IP transfers only.

Range: 1 to 9999

Default: 10

UNIT

IBM Mainframe File Creation parameter

Specifies the unit on which to create the file. Ignored for files created on the Tandem.

Range: Up to six characters

Default: None

UNITCT

Specifies the number of units to be allocated on the remote system. This is a tape parameter and is used when the partner is an IBM mainframe.

Range: 1 to 20

Default: None

USERID

Identifies the remote user ID for use with the file security scheme on the remote system.

Range: Up to 12 characters

Default: None

VERSION

Locally initiated transfers only.

Indicates whether the request is a Version 1 or Version 2 transfer.

1

Version 1

2

Version 2

Default: 2

VOLCT

Specifies the maximum number of volumes to be used in processing a multi-volume output tape data set on the remote system.

Range: 1 to 255

Default: None

VOLSQ

Specifies the sequence number of the first volume of a multi-volume remote data set to be used.

Range: 1 to 255

Default: None

VOLUME

IBM Mainframe File Creation parameter

Specifies the volume on which to create the file.

Range: Up to six characters

Default: None

XBUFFSIZE

Specifies the buffer size for a single record. Set this to the maximum record size for the transfer.

For HP NonStop records, the maximum record size is 4096.

Range: 0000 to 4096

Default: 4096

XCOM_SHOW_CIPHER

Specifies whether to display encryption algorithms in the CA XCOM Data Transport queue detailed information, which is used for transfers.

NO

Do not display encryption algorithms in the queue detail information.

YES

Display encryption algorithms in the queue detail information.

Default: NO

XCOM_CONFIG_SSL

Specifies the name of the SSL configuration file. Use the following format:

vol.subvol.filename

Range: Up to 27 characters.

Default: XCSSLCNF

Note: The value for XDIR will be used if specified.

XDIR

Specifies the default volume and subvolume for all files read and written by CA XCOM Data Transport for HP NonStop except the XCOMCNF, XCOMHIST, XCOMPWF, and CKPTFIL files.

Note: If a transfer is initiated remotely and this parameter is not specified, CA XCOM Data Transport uses the default volume of the user ID that the remote system sends.

Range: Up to 256 characters

Default: None

XIDEST

Indicates the intermediate destination name for indirect transfers. For SNA, the mainframe LU name must be configured in SNAX/APC.

If XIDEST is null or unset, a direct connection will be attempted to the remote system.

If XIDEST contains a value, it is taken to be the name of an intermediate CA XCOM Data Transport destination which will handle traffic to and from the named remote system.

Range: Up to 21 characters

Default: None

XLOG_FILE_TYPE

Controls which Guardian file type will be created.

Since multiple processes will be writing simultaneously to the entry sequence files, the process name will be provided in addition to the time stamp. If it is an unnamed process, the PID will be used.

If the log/trace file already exists, it ignores this parameter.

Range: EDIT and ENTRYSEQ

Default: EDIT

XLOGFILE

For locally initiated CA XCOM Data Transport transfers only.

Provides a file name for the log file for locally initiated transfers.

Range: Up to 250 characters.

Default: XCOMLOG

XLUNAME

Identifies the local LU name to use for SNAX.

Range: Up to eight characters

Default: None

XMODE

Identifies the SNA LOGMODE for SNAX.

Range: Up to eight characters

Default: XCOMMODE

XQUE_FILE

Specifies a prefix that to be used, along with the HP NonStop spool files seven-character job number prefixed with a J, to build a unique REMOTE_FILE file name.

This process is overridden if a value for the REMOTE_FILE parameter is specified.

Note: For xque send file transfers only

Range: Up to 248 characters

Default: None

XTRACE

Indicates the trace level.

Range: 0 to 9

0

No tracing.

9 (maximum)

Outputs the raw contents of data buffers.

Important! Because non-ASCII characters may be interpreted as control characters on your terminal, use the highest trace level with caution.

Default: 0

Appendix B: Messages

CA XCOM Data Transport for HP NonStop includes a set of error messages. These messages are written to a specified log file on the local system (see the XLOGFILE parameter). Some messages are also displayed on the user's terminal and/or sent to the remote system.

This section contains the following topics:

Message Format

The general format of a CA XCOM Data Transport message is as follows:

```
system_identifier message_number message_type  message
```

Parts of the Message

The parts of a CA XCOM Data Transport message are as follows:

XCOM (positions 1 to 4; alphabetic)

Displays the first four characters of a CA XCOM Data Transport message.

system identifier (position 5; alphabetic)

Specifies the CA XCOM Data Transport system that generated the message. Valid values include the following:

- A—Gateway
- D—OpenVMS Alpha
- E—IBM z/VSE
- K—IBM CICS
- M—IBM z/OS
- N—Windows
- R—Netware
- S—IBM i5/OS (AS/400)
- T—HP NonStop (Tandem)
- U—UNIX and Linux
- V—IBM z/VM
- 8—Stratus Computer

message no (positions 6 to 9; numeric)

Specifies the message number.

message type (position 10; alphabetic)

Indicates the message type, as follows:

- I—An informational message. No action is required on the part of the user.
- E—An error message. Usually some action is necessary to correct the problem or to determine the cause.

Notes:

- Messages with numbers ranging from 0 to 255 are informational or prompt messages.
- Messages numbering between 100 and 255 represent CA XCOM Data Transport states and are used in traces and so on.
- Message numbers of 256 and above are error messages.

message text (alphanumeric)

Displays the message text.

Message Examples

The following is an example of a CA XCOM Data Transport for HP NonStop message:

```
XCOMT0011I XCOM62 ENDING TRANSFER
```

The following is an example of a CA XCOM Data Transport for PC information message:

```
XCOMC0000I DATA TRANSFER SUCCESSFUL
```

List of Messages

This section contains CA XCOM Data Transport messages.

0007I

Error code: 0 Descriptive message

Reason:

This is a return code from the underlying system software. It is usually followed by a message describing the meaning of the code.

Action:

Refer to the documentation for the SNA LU 6.2 or other system software.

0010I

Starting XCOM Transfer on Tue Mar 17 14:45:59 1992

Reason:

This start of transfer message is issued when a locally initiated transfer is begun. It is simultaneously placed in the transfer log.

Action:

None required.

0011I

Ending Transfer

Reason:

This message marks the completion of a locally initiated transfer. It placed in the log and written to standard error at the end of a transfer.

Action:

None required.

0024I

Transfer scheduled for future execution.

Reason:

The transfer has been scheduled for future execution. The date and time for initiating the transfer were specified in a configuration file or from the user interface.

Action:

None required.

0028I

Starting locally initiated transfer.

Reason:

Locally initiated transfer starting.

Action:

None required.

0029I

Locally initiated transfer started.

Reason:

Locally initiated transfer started.

Action:

None required.

0052I

Transfer held.

Reason:

The transfer was held from starting at the time for which it was scheduled.

Action:

None required.

0053I

Transfer released.

Reason:

The transfer released to start at the time for which it was scheduled.

Action:

None required.

0103I

TP_VALID

Reason:

CA XCOM Data Transport is about to enter TP_VALID state. In this state, the CA XCOM Data Transport accepts incoming remote allocates if the GET_ALLOCATE verb has been issued.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0105I

ALLOCATE_CONVERSATION

Reason:

CA XCOM Data Transport is about to enter ALLOCATE_CONVERSATION state. It is about to issue the allocate verb for the remote transaction program.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0106I

GET_ALLOCATE

Reason:

CA XCOM Data Transport is about to enter GET_ALLOCATE state. In this state, incoming allocates are sought by the local transaction program.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0107I

DEALLOCATE_CONVERSATION

Reason:

CA XCOM Data Transport is about to enter DEALLOCATE_CONVERSATION state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0108I

SEND_HEADER

Reason:

CA XCOM Data Transport is about to enter SEND_HEADER state. In this state, a buffer is allocate, the CA XCOM Data Transport header record is created from the transfer parameters and the header is sent.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0109I

REQUEST_HEADER_CONFIRMATION

Reason:

CA XCOM Data Transport is about to enter REQUEST_HEADER_CONFIRMATION state where the incoming header is confirmed and, if necessary, the connection is turned around

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0111I

SEND_MAXLRECL

Reason:

CA XCOM Data Transport is about to enter SEND_MAXLRECL state. Here CA XCOM Data Transport will send the maximum logical record length as specified in the configuration file.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0112I

SENDING_DATA

Reason:

CA XCOM Data Transport is about to enter SENDING_DATA state. CA XCOM Data Transport sends one data record each time it enters this state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0113I

DATA_CONFIRM

Reason:

CA XCOM Data Transport is about to enter DATA_CONFIRM state. In this state, the transaction program issues the confirm that is sent at the end of a data file.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0114I

SEND_TRAILER

Reason:

CA XCOM Data Transport is about to enter SEND_TRAILER state. The trailer record containing the number of records is sent.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0115I

TRAILER_CONFIRM

Reason:

CA XCOM Data Transport is about to enter TRAILER_CONFIRM state. The CA XCOM Data Transport transaction program is about to issue an deallocate confirm verb for the conversation.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0116I

RECEIVE_HEADER

Reason:

CA XCOM Data Transport is about to enter RECEIVE_HEADER state. In this state, CA XCOM Data Transport receives the incoming header.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0117I

CONFIRM_HEADER

Reason:

CA XCOM Data Transport is about to enter CONFIRM_HEADER state. The CA XCOM Data Transport Transaction program is about to issue the confirmed LU 6.2 verb.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0118I

RECEIVE_MAXLRECL

Reason:

CA XCOM Data Transport is about to enter RECEIVE_MAXLRECL state. The transaction program receives the maximum record length record and uses it to initialize system parameters.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0119I

RECEIVE_DATA

Reason:

CA XCOM Data Transport is about to enter RECEIVE_DATA state. In this state, the data records are received, decompressed, unpacked, and written to the target file.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0120I

DATA_CONFIRMED

Reason:

CA XCOM Data Transport is about to enter DATA_CONFIRMED state. It issues the confirmed verb and closes the received file in preparation for receiving the trailer record.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0121I

RECEIVE_TRAILER

Reason:

CA XCOM Data Transport is about to enter RECEIVE_TRAILER state. A receive_and_wait verb is issued for the incoming CA XCOM Data Transport trailer record.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0122I

TRAILER_CONFIRMED

Reason:

CA XCOM Data Transport is about to enter TRAILER_CONFIRMED state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0123I

PROCESS_DATA

Reason:

CA XCOM Data Transport is about to enter PROCESS_DATA state. In this state, the transaction program compares the record count actually received to the count sent in the trailer record. If they don't match, CA XCOM Data Transport enters an invalid trailer state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0125I

TERMINATE_INITIATE_CHECK

Reason:

CA XCOM Data Transport is about to enter TERMINATE_INITIATE_CHECK state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0127I

TP_DONE

Reason:

CA XCOM Data Transport is about to enter TP_DONE state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0152I

LOCAL_SEND

Reason:

CA XCOM Data Transport is about to enter LOCAL_SEND state. In this state, the transaction program initializes several internal variables, logs the startup message, and sets the userid for this transfer.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0153I

LOCAL_RECEIVE

Reason:

CA XCOM Data Transport is about to enter LOCAL_RECEIVE state. In this state, the transaction program initializes several internal variables, logs the startup message, and sets the userid for this transfer.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0154I

REMOTE_SEND

Reason:

CA XCOM Data Transport is about to enter REMOTE_SEND state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0155I

REMOTE_RECEIVE

Reason:

CA XCOM Data Transport is about to enter REMOTE_RECEIVE state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0156I

OPEN_REMOTE_INPUT_FILE

Reason:

CA XCOM Data Transport is about to enter OPEN_REMOTE_INPUT_FILE state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0157I

OPEN_LOCAL_INPUT_FILE

Reason:

CA XCOM Data Transport is about to enter OPEN_LOCAL_INPUT_FILE state. The local input file is about to be opened.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0158I

OPEN_OUTPUT_FILE

Reason:

CA XCOM Data Transport is about to enter OPEN_OUTPUT_FILE state. The output file is about to be opened.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0159I

LOCAL_ATTACH

Reason:

CA XCOM Data Transport is about to enter LOCAL_ATTACH state. In this state, the transaction program tries to establish a connection to the local LU 6.2 SNA server.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0161I

REMOTE_ATTACH

Reason:

CA XCOM Data Transport is about to enter REMOTE_ATTACH state. In this state, the invoked transaction program tries to establish a connection to the local LU 6.2 SNA server.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0162I

SET_UP_OVERLAY

Reason:

CA XCOM Data Transport is about to enter SET_UP_OVERLAY state. In this state, the indirect transfer header record is created.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0163I

SEND_OVERLAY

Reason:

CA XCOM Data Transport is about to enter SEND_OVERLAY state. In this state, it sends the overlay record that is part of the indirect transfer protocol.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0164I

RECEIVE_OVERLAY

Reason:

CA XCOM Data Transport is about to enter RECEIVE_OVERLAY state. An indirect transfer has been requested by the remote side and the overlay record is about to be received.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0165I

DO_SYSTEM

Reason:

CA XCOM Data Transport is about to enter DO_SYSTEM state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0167I

SET_REMOTE_USER_ID

Reason:

CA XCOM Data Transport is about to enter SET_REMOTE_USER_ID state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0168I

DO_COMMAND

Reason:

CA XCOM Data Transport is about to enter DO_COMMAND state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0180I

SENDING_ERROR

Reason:

CA XCOM Data Transport is about to enter SENDING_ERROR state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0181I

DISPLAY_ERROR

Reason:

CA XCOM Data Transport is about to enter DISPLAY_ERROR state. The error is displayed on the local display.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0182I

RECEIVE_ERROR

Reason:

An error message is being received from the remote partner. The CA XCOM Data Transport transaction program issues a receive and wait verb to retrieve the text.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0183I

LOG_ERROR

Reason:

CA XCOM Data Transport is about to enter LOG_ERROR state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0184I

LOG_COMPLETE

Reason:

CA XCOM Data Transport is about to enter LOG_COMPLETE state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0185I

SEND_ERROR_MESSAGE

Reason:

CA XCOM Data Transport is about to enter SEND_ERROR_MESSAGE state. A send error verb has been issued and the text of the error encountered is sent in this state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0186I

WAIT_FOR_REMOTE_DEALLOCATE

Reason:

CA XCOM Data Transport is about to enter WAIT_FOR_REMOTE_DEALLOCATE state.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0187I

TRANSFER_FAILED

Reason:

The CA XCOM Data Transport transfer has failed.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0188I

CHECK_RETURN_CODES

Reason:

CA XCOM Data Transport is checking return codes at the end of a transfer.

Action:

None. This message is used in the CA XCOM Data Transport trace to report the state of the CA XCOM Data Transport transaction program.

0283E

Error allocating send buffer

Reason:

Error encountered when trying to increase send buffer size using realloc() call.

Action:

Retry transfer. This error may be caused by memory fragmentation.

0286E

Error setting local user id

Reason:

Error setting local user id. Call to setuid(getuid()) failed.

Action:

Be sure that the local CA XCOM Data Transport transaction program is either owned by the invoking user or is running setuid root.

0287E

Error setting remote user id

Reason:

An error was encountered while trying to set the remote user id.

Action:

Check that the remotely requested user id is valid on your system and that the remote initiator has entered it and the password for it correctly.

0288E

System function failed

Reason:

A call to system() has failed. This function is used to initiate command functions from within CA XCOM Data Transport.

Action:

Retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0289E

Command failed

Reason:

A command issued using the system() call has returned a non-zero completion code.

Action:

Retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0290E

Receive of indirect transfer record failed

Reason:

An error has occurred while attempting to receive the overlay record for an indirect file transfer.

Action:

Check SNA connectivity and retry. If the problem persists, contact CA XCOM Data Transport Technical Support.

0291E

Send of indirect transfer record failed

Reason:

An error has occurred while attempting to send the overlay record for an indirect file transfer.

Action:

Check SNA connectivity and retry. If the problem persists, contact CA XCOM Data Transport Technical Support.

0292E

Error sending error message to remote system

Reason:

An error was encountered while sending an error message to remote system.

Action:

Check SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0296E

Error deallocating conversation

Reason:

Error detected while deallocating a send conversation. This deallocate occurs after sending a file.

Action:

Check SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0297E

Error requesting header confirmation

Reason:

The remote CA XCOM Data Transport transaction program did not confirm the transfer request header that it received

Action:

Check SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0298E

Unable to allocate remote transaction program

Reason:

The APPC interface returned an error after an attempt was made to allocate a CA XCOM Data Transport transaction program on a remote LU.

Action:

Check that the remote transaction program can be allocated. On MVS, this means that the APPL is active. On other systems, there are different criteria for accessibility.

0299E

Cannot attach to LU 6.2 facility on local initiate

Reason:

An error was detected when a locally initiated transfer request attempted to attach to the APPC manager process.

Action:

Check SNA status as well as connection status. Check the REMOTE_SYSTEM parameter to make sure that it contains the name of the connection profile that you want to use for this transfer.

0300E

Cannot attach to LU 6.2 facility on remote allocate

Reason:

An error was detected when a remotely initiated transfer request transaction program attempted to attach to the APPC manager process.

Action:

Check SNA configuration and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0301E

Unable to Enable APPC Communications

Reason:

This message will not occur when using SNA services on the IBM RISC System/6000.

Action:

None required.

0302E

Unable to open local input file: Error 0

Reason:

A problem was encountered while trying to open a CA XCOM Data Transport input file; the data following the explanation is the sense.

Action:

Check the existence and permissions on the file you have specified.

0303E

Send of XCOM header failed

Reason:

An error has occurred while attempting to send a transfer header record.

Action:

Check SNA connectivity and retry. If the problem persists, contact CA XCOM Data Transport Technical Support.

0304E

Send of Maximum Record Length failed

Reason:

An error has occurred while attempting to send the maximum logical record length as part of the file transfer setup procedure.

Action:

Check SNA connectivity and retry. If the problem persists, contact CA XCOM Data Transport Technical Support.

0305E

Receive of incoming header failed

Reason:

An error was detected while attempting to receive a transfer request header from a remote system.

Action:

Check SNA connectivity and retry. If the problem persists, contact CA XCOM Data Transport Technical Support.

0307E

Error Invalid PACK OPTION

Reason:

CA XCOM Data Transport for HP NonStop does not support the PACK option.

Action:

Use a different option or approach.

0309E

Error reading local input file : Error 0

Reason:

Error reading input file. System call read() returned -1. The error number displayed is the value of the system variable errno.

Action:

Check the sense code on the message, fix the problem, and retry the transfer.

0310I

Received error from remote system.

Reason:

An error has been received from the remote system and has been placed in the log.

Action:

Check the error received from the remote system and fix the problem detailed there. Errors messages from partner CA XCOM Data Transport systems can be found in their manuals.

0311E

Send of user data record failed.

Reason:

An error was returned from the APPC interface while attempting to send data to a remote system.

Action:

Check SNA connectivity and retry. If the problem persists, contact CA XCOM Data Transport Technical Support.

0312E

Confirm to last data record failed.

Reason:

The APPC interface has returned an error when attempting to issue a confirm following the last data record in a transfer.

Action:

Check SNA connectivity and retry. If the problem persists, contact CA XCOM Data Transport Technical Support.

0313E

Negative response to data confirm request.

Reason:

Negative response to data confirm request.

Action:

Contact CA XCOM Data Transport Technical Support.

0314E

Send of trailer record failed.

Reason:

Send of trailer record failed. An error was returned by the SNA subsystem after the CA XCOM Data Transport trailer was sent.

Action:

Check SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0315E

Trailer record not confirmed

Reason:

Trailer record not confirmed. The remote side has detected an error in the CA XCOM Data Transport trailer, either in its format or in the number of records transferred.

Action:

This error may be the result of send a text file which contains zero-length records. If this is the case, try to send the file with compression turned on.

0319E

Error confirming checkpoint

Reason:

There is an error in confirming the checkpoint.

Action:

Check the SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0320E

Error issuing confirmed for checkpoint

Reason:

There is an error issuing confirmed for the checkpoint.

Action:

Check the SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0321E

Unable to open checkpoint file

Reason:

There is an error opening the checkpoint file.

Action:

Check sense code on message, fix problem and retry the transfer.

0322E

Unable to write to checkpoint file

Reason:

There is an error writing to the checkpoint file.

Action:

Check sense code on message, fix problem and retry the transfer.

0403E

Cannot open output file: Error 0

Reason:

Cannot open output file using requested action.

Action:

Check sense code on message, fix problem and retry the transfer.

0404E

Error confirming header

Reason:

Error encountered when attempting to issue confirmed response to incoming header.

Action:

Check SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0405E

Error receiving maxreclen.

Reason:

An error was encountered when trying to read an incoming maximum record length record or when trying to realloc() a read buffer of the size specified in the record received.

Action:

Check SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0406E

Error while trying to read Feature Negotiation Record

Reason:

An error was encountered while trying to read a feature negotiation record.

Action:

Check the SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0408E

Error encountered in feature negotiation protocol

Reason:

An error was encountered in feature negotiation protocol.

Action:

Check the SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0409E

Error encountered trying to confirm feature negotiation record

Reason:

An error was encountered in trying to confirm a feature negotiation record.

Action:

Check the SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0411E

Error while confirming feature negotiation record

Reason:

An error occurred while confirming a feature negotiation record.

Action:

Check the SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0412E

Error receiving data

Reason:

Error detected by the APPC receive and wait verb while reading incoming data.

Action:

Check SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0413E

Trailer invalid

Reason:

The count in the received trailer record did not match the actual number of records received.

Action:

Check SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0415E

Error receiving trailer

Reason:

The receive and wait issued for an incoming trailer record has failed.

Action:

Check SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0416E

Error writing output file

Reason:

The write() system call has returned a -1. This probably means that the file system is out of space.

Action:

Check the free space in the file system.

0418E

Transmission interrupted: signal received

Reason:

An interrupt signal as been received and transmission has been interrupted.

Action:

Retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0423E

Error receiving file descriptor

Reason:

An error occurred while trying to receive a file descriptor record.

Action:

Check the SNA connectivity and retry the transfer. If the problem persists, contact CA XCOM Data Transport Technical Support.

0424E

Attempting to create an existing file

Reason:

Attempting to create an existing file.

Action:

Decide whether you want to replace the target file or to create one using a different name. Set parameters appropriately and retry transfer.

0425E

Cannot position file on a restart

Reason:

When trying to reposition a file to restart a transfer, the fseek() call returned a -1. This usually indicates a file that has been updated since its transfer was interrupted.

Action:

Determine why the file was updated and/or retry the transfer.

0437E

Record larger than input buffer.

Reason:

The length of a file record is greater than the length of the input buffer.

Action:

Adjust size of buffer and retry.

0438E

Record length greater than maxreclen, but truncation not allowed.

Reason:

The length of the received record is greater than maxreclen, but truncation is not allowed.

Action:

Adjust size of buffer and retry.

0440E

Internal buffer maxreclen received.

Reason:

Internal buffer is smaller than maxreclen received.

Action:

Adjust size of buffer and retry.

0449E

Type of compression not supported

Reason:

An unsupported type of compression was specified in the header.

Action:

Retry the transfer using compression type YES, NO, COMPACT, COMPACTL, or RLE.

0468E

Cannot allocate memory.

Reason:

Cannot allocate memory.

Action:

Retry the operation. If the problem persists, contact CA XCOM Data Transport Technical Support.

0471E

Invocation mode undefined.

Reason:

Internal error. Invocation mode undefined.

Action:

Retry the operation. If the problem persists, contact CA XCOM Data Transport Technical Support.

0472E

Current mode undefined.

Reason:

Internal error. Current mode undefined.

Action:

Retry the operation. If the problem persists, contact CA XCOM Data Transport Technical Support.

0488E

Input file not specified, but stdin cannot be used.

Reason:

Local input file is not specified, but stdin cannot be used.

Action:

Specify an input file and retry.

0489E

Output file not specified: stdout cannot be used.

Reason:

Local output file is not specified, but stdout cannot be used.

Action:

Specify an output file and retry.

0530E

Record exceeds IO_BUFFSIZE

Reason:

The length of the compressed buffer exceeds the I/O buffer size.

Action:

Do one of the following:

- Adjust IO_BUFFERSIZE.
- Select a different compression.
- Set COMPRESS=NO.

0535E

Error compress flag in feature negotiation protocol.

Reason:

Compression specified is not supported.

Action:

Specify a valid compression type.

0785I

Starting TCP/IP Connection

Reason:

A TCP/IP connection is being started to a remote partner.

Action:

None

0786I

TCP/IP Connection Established

Reason:

A TCP/IP connection was successfully made to a remote partner.

Action:

None

0793I

Remote TCP/IP Connection Established

Reason:

A TCP/IP remote connection was successfully established from a remote partner.

Action:

None

0805I

TCP/IP Connection Ended

Reason:

A TCP/IP connection completed.

Action:

None

0811I

Starting Secure TCP/IP Connection

Reason:

A Secure TCP/IP connection is being started to a remote partner.

Action:

None

0812I

Remote Secure TCP/IP Connection Requested

Reason:

A request for a Secure TCP/IP remote connection was received from a remote partner.

Action:

None

0813I

Secure TCP/IP Handshake Complete

Reason:

Secure Socket (SSL) negotiation for the connection successfully completed. The connection is now a Secure TCP/IP connection.

Action:

None

0814I

Secure TCP/IP Connection Requested

Reason:

A request for a Secure TCP/IP connection was sent to the remote partner.

Action:

None

0818I

Secure TCP/IP Connection Ended

Reason:

A Secure TCP/IP connection completed.

Action:

None

Appendix C: CA Problem Determination

This appendix contains information on general procedures to use for determining problems and worksheets to assist you in problem determination. CA XCOM Data Transport is supported on different operating systems on a variety of platforms. With each implementation, the problem determination procedures vary slightly.

This section contains the following topics:

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Knowledge Requirements

This appendix is focused towards help desk personnel or a designated CA XCOM Data Transport troubleshooter with knowledge of the following:

- The operational characteristics of CA XCOM Data Transport for HP NonStop
- For transfers that use SNA, some knowledge of the structure and components of an SNA networking environment, including an understanding of the nature of SNA LUs, PUs, sessions, and conversations.

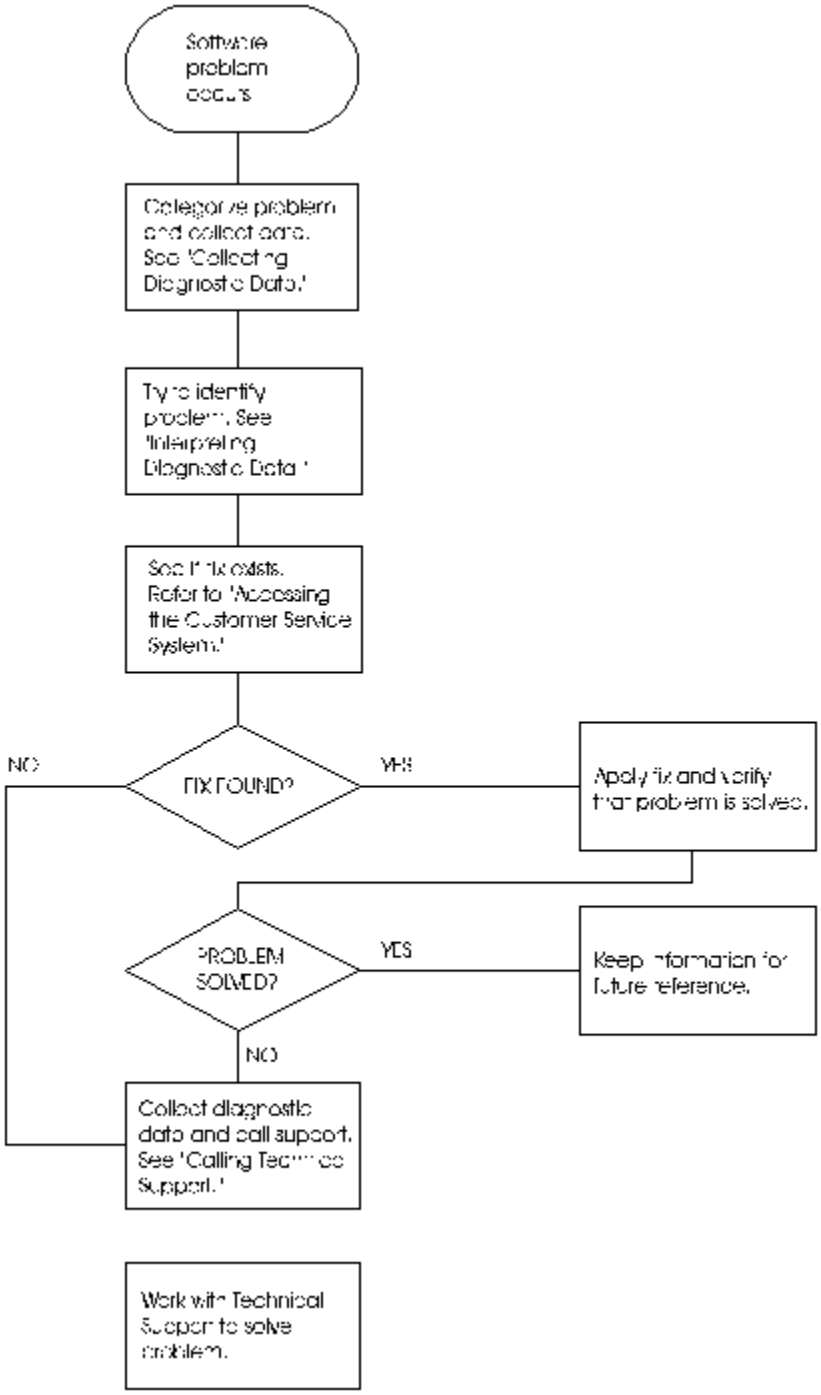
The majority of problems encountered by users is a result of the following:

- Improper configuration
- Incorrectly performed procedures
- A faulty environment

The purpose of the procedures and worksheet in this appendix is to help you to isolate the cause of problems and to correct them in an efficient and timely manner or to gather enough information for CA XCOM Data Transport Technical Support to help you as quickly as possible.

Refer to the following "Diagnostic Procedures" flowchart for a summary of the procedures you should follow if you have a problem with a CA software product. Each of these procedures is detailed in this appendix.

Diagnostic Procedures



Collect Diagnostic Data

To be effective, you must follow these procedures in order and document the results of every test carefully. It is good practice to run through the tests at least twice to ensure that the results are consistent.

1. Document the precise actions that were performed immediately prior to the appearance of the problem.

   Note: The description of the actions performed should be detailed enough to allow for you to recreate the problem at some future time if necessary.

2. Document the symptom(s) of the problem, including the following:

   - Error messages
   - Unexpected events
   - Other details pertinent to the malfunction.

   Wherever possible, record any error messages that appear on the screen.

3. If a data file has been corrupted, save the data file in a safe place for later analysis by CA XCOM Data Transport Technical Support. Any information that appears even remotely relevant may prove vital in achieving a quick resolution to the problem.

4. Retry the operation, isolating and eliminating as many extraneous factors as possible.

5. Did the problem recur? Use the answer to determine your next step, as follows:

   No

   Record as much as you can from the problem's first occurrence (log files, and so on) so that if you ever see it again, you will have additional evidence.

   Yes

   Continue with Step 6.

6. Is an error message issued?

   Yes

   Look up the error message in the appropriate CA XCOM Data Transport manual and follow the instructions.

   No

   If no error message was issued and the operation does not complete (for example, hangs), the problem may be related to a system malfunction (for example, remote system response time problems, and so on).

   Leave the function in the "hung state" for at least 10 minutes. If the problem persists, contact a person who can verify the current status of the HP NonStop system and backbone network. If the HP NonStop system or the network is not operating normally, retry the transfer at some later time.

7. Run a trace to pinpoint and trace problems in your communication lines.

General Methodology

The following is a step-by-step methodology for isolating and resolving problems. Do each step in order and carefully document the results of every test. It is good practice to run through the tests at least twice, in order to ensure consistent results.

1. Document previous actions.
2. Document symptoms.
3. Recreate the problem.
4. Look for error messages.

Document Previous Actions

Write down everything you did just before the problem appeared. Be detailed enough to allow for the problem to be recreated.

Document Symptoms

Write down any symptoms of the problem. Include error messages, unexpected events, or other details pertinent to the malfunction. Wherever possible, use the system's Print Screen capability to record any error messages that appear on the screen. If a data file has been corrupted, save the data file in safe place for later analysis by CA. Any information that appears even remotely relevant may prove vital in achieving a quick resolution to the problem.

Recreate the Problem

Try to recreate the problem. If the operation works correctly, this is an intermittent problem. Try to isolate the cause of the problem at the times when it occurs by using a trace. Procedures for doing this are outlined in the remainder of this appendix.

Take steps to determine the frequency of the problem and the external events that cause it. This will involve retrying the operation several times at various times of the day and recording the success or failure rate. Try to correlate the failures with some external event (for example, failure only occurs during peak business hours or only after some other activity, and so on).

When you are confident that the problem can be reproduced with reasonable frequency, analyze the relationship between the offending external event and the nature of the CA XCOM Data Transport failure. Contact CA Technical Support to discuss further investigative action or problem resolution.

Look for Error Messages

If an error message appears as a result of your operation, look up that error message in the appropriate CA XCOM Data Transport manual and follow the instructions there. Refer to page C-4 (???) for notes on CA XCOM Data Transport error messages.

If no error message was issued and the operation does not complete (that is, it hangs), the problem may be related to a system malfunction (for example, HP NonStop failure, remote system response time problems, and so on). Leave the function in the hung state for at least 10 minutes. If the problem persists, contact a person who can verify the current status of the HP NonStop and backbone SNA network. If the HP NonStop system or the network is not operating normally, retry the transfer at some later time.

Problem Determination Worksheet

This section contains worksheets for use in determining problems. When you call CA XCOM Data Transport Technical Support, it is helpful to have the following information available so they can quickly and efficiently narrow down the cause of the problem. Use the following sheets to gather the information Technical Support will need.

General Information

Use the following table to provide general customer information:

Information	Your Response
Date of Incident	
Incident Number (to be supplied by CA XCOM Data Transport Technical Support)	
Customer Name	
Customer Contact	
Customer Address	
Customer Telephone Number	
Customer Fax Number	
CA XCOM Data Transport Platform, Version, Release, and Maintenance	
Technical Support Contact (to be supplied by CA XCOM Data Transport Technical Support)	

Environment Information

Use the following table to inventory information about your environment. Use additional sheets as necessary.

Information Type	Initiating System	Receiving System	(Receiving System)	Intermediary System
Platform				
Operating System				
Communications Subsystem				
Security Subsystem				
For SNA/APPC (Version/ Release/ Maintenance)				
Have there been any hardware or software changes (for example, hardware upgrades, operating system, CA XCOM Data Transport or communications type changes)				

Transfer Type

Please check the box that applies to the file transfer:

Send File	Send Job	Send Report	Retrieve File

Problem Description

What is the customer attempting to perform or accomplish?

Provide a detailed problem description:

Use the following table to answer the questions below:

Question	YES	NO
Is the problem reproducible?		
Is this an immediate transfer (Foreground, TYPE=EXECUTE) or a scheduled transfer (Background, TYPE=SCHEDULE)?		
Has there been any hardware or software changes (for example, hardware upgrades, operating system, CA XCOM Data Transport or communications type changes)		

If YES, please explain in detail.

Problem History

Use this section to provide a description of the problem history:

	Yes	No
Is this the first occurrence of the problem?		

If No, please explain.

Error Messages

Please provide a list of error messages; include the message ID and all of the text.

Network Configuration Diagram

Draw your network configuration diagram on this page. Illustrate how the partners in your network are connected.

Environmental Information Inventory

Use the following table to inventory information about your environment. Fill in the boxes of platforms involved in the transfer:

Platform	Operating System	Communications Subsystem	Security Subsystem	APPC Version/ Release/ Maintenance
z/OS				
z/VM				
z/VSE				
AS/400				
VAX				
Stratus				
HP NonStop				
HP/9000				
RS/6000				

Sun Solaris
OS/2
DOS
Windows
Windows NT
NetWare
Other

Initiating Platform _____

Please check the box that applies to the file transfer:

	Send File	Send Job	Send Report	Receive File
This is a				

Documentation

The following is a starting point for various error conditions on various platforms. More documentation may be required to resolve the error condition:

Conditions	Platforms	Documentation
Abend	z/OS	Full Dump JES2 or JES3 log Xcom log Batch job stream or ISPF parameters
	z/VM	GCS Dump Virtual machine Console log Dest Table Default File CA XCOM Data Transport parameter file
	z/VSE	Partition DUMP,POWER log CA XCOM Data Transport log Batch Job stream Dest and Default Tables
	HP NonStop	Abend message Save abend file (ZZSAnnnn, file code 130) What was being attempted Transfer parameters VPROC of XCOM62

Conditions	Platforms	Documentation
Error in Transfer	z/OS	JES2 or JES3 log CA XCOM Data Transport log CA XCOM Data Transport trace Batch Job Stream or ISPF parameters Xcomcntl and default file VTAM Buffer/IO trace
	z/VM	VTAM Buffer/IO trace CA XCOM Data Transport trace Virtual Machine Console log Rexx exec or Clist Dest and default file
	z/VSE	POWER Log CA XCOM Data Transport log CA XCOM Data Transport trace Batch Job Stream VTAM Buffer/IO trace Dest and default tables
	AS/400	Error message received Sense codes Job logs
	OS/2	Comm. Manager Trace (APPC only) CA XCOM Data Transport log CA XCOM Data Transport parameter file
	DOS	Tptrace CA XCOM Data Transport log CA XCOM Data Transport parameter file
	HP NonStop	SNA Trace CA XCOM Data Transport trace CA XCOM Data Transport Config Parameters used for transfer Local initiator log Remote initiator log

Conditions	Platforms	Documentation
	VAX	SNA Trace CA XCOM Data Transport Config Parameters used for transfer Local initiator log Remote initiator log
	Stratus	Syserr_log messages Network definitions on both sides of the connection Monitor_SDLC log Status of station, link, line, LU and session
	NetWare	Syserr_log messages Network definitions on both sides of the connection Monitor_SDLC log Status of station, link, line, LU and session
	Other	

Run a Trace

There are many tracing options available on CA XCOM Data Transport for HP NonStop.

Standard CA XCOM Data Transport Trace

This trace covers all aspects of CA XCOM Data Transport while running. It includes CA XCOM Data Transport state transitions in the format shown in the appendix "Messages." It also shows buffer contents, fatal and non-fatal errors, parameter validation information, file opens, and so on.

You can produce this trace by setting the XTRACE parameter (see the appendix "Parameters"). This trace is sent either to local standard output or the remote trace file (RT*yymmdd*).

The CA XCOM Data Transport Trace and Log Facility

The basic documentation needed to resolve any problems is the CA XCOM Data Transport trace. CA XCOM Data Transport tracing is controlled by the XTRACE parameter in the XCOMCNF file. XTRACE should be the first line in the XCOMCNF file. The tracing facility imposes significant overhead, so it should only be used when determination is necessary.

XTRACE Level	Output
0	nothing written
1	error messages
2	CA XCOM Data Transport initialization
3	more details
4	start and end of transfer
5	more details
6	more details
7	middle of transfer, events that occur on a record basis
8	more details
9	more details

For locally initiated file transfers, the trace is written to STDERR, which is normally directed to the terminal. It can be redirected to a file using the ASSIGN command:

```
ASSIGN STDERR,filename
```

For remotely initiated transfers using SNAX, CA XCOM Data Transport uses the SNA LU name as the subvolume for the trace file. For remotely initiated TCP/IP transfers, trace files are written to the subvolume CAXCOM. The volume is taken from the XDIR parameter in the XCOMCNF file. The file name is RT followed by the date.

Example:

```
RT970915
```

The CA XCOM Data Transport local log, remote log, and remote trace files can be written to entry sequence files, so that multiple CA XCOM Data Transport processes can write to the same log/trace file. The CA XCOM Data Transport parameter XLOG_FILE_TYPE controls which Guardian file type is created. The valid values are EDIT and ENTRYSEQ. The default value is ENTRYSEQ. If the log/trace file already exists, it ignores this parameter. Because multiple processes write simultaneously to the entry sequence files, the process name is provided in addition to the time stamp. If it is an unnamed process, the PID is used.

The RTRACE and RLOGFILE parameters control the name and locations of remote trace and log files. You can specify as little of the path name as you like. For example, if CA XCOM Data Transport creates the remote log file name as $DSV.CAXCOM.RL970915, you can specify the following RLOGFILE values to create the file names shown:

$SYSTEM

$SYSTEM.CAXCOM.RL970915

$SYSTEM.XCOM

$SYSTEM.XCOM.RL970915

$SYSTEM.XCOM.RLOG

$SYSTEM.XCOM.RLOG

If you only specify volume or volume/subvolume, a new file name is created when the day changes (assuming the XCOM process is still up and RLOG_SECURITY is set to NO).

RTRACE, RLOGFILE, and XLOGFILE can all be set to NONE. If NONE, no file is created and no logging or tracing information is written.

SNA Traces

To use the SNAX/APC APCCOM trace for SNA traces

1. Run the program APCCOM to start and stop traces.
2. Use the program APCTAP to examine traces.

If there is no information in the trace file, you have a configuration problem on HP NonStop.

If the BIND fails, you should see error (sense-data) information sent by the remote system. To determine the cause of the error, see your *IBM Formats* manual.

Interpreting Diagnostic Data

When you have collected the specified diagnostic data, write down your answer to the following questions:

1. What was the sequence of events prior to the error condition?
2. What circumstances existed when the problem occurred and what action did you take?
3. Has this situation occurred before? What was different then?
4. Did the problem occur after a particular PTF was applied or after a new release of the software was installed?
5. Have you recently installed a new release of the operating system?
6. Has the hardware configuration (tape drives, disk drives, and so on) changed?

From your response to these questions and the diagnostic data, try to identify the cause and resolve the problem.

CA XCOM Data Transport Error Messages

If an error message is generated during a CA XCOM Data Transport transfer, that error message is logged on the system on which the error occurred and also transported to the other system involved in the transfer. For example, if an HP NonStop program is attempting to transfer a file to a z/OS system where the allocated space is insufficient, then a message is generated by the z/OS component of CA XCOM Data Transport, logged in the z/OS log, and reported to the HP NonStop system.

Note: On the HP NonStop system, these messages can be viewed only if the programmer has turned notification on to allow for the messages to be displayed.

System Codes

The 5th character of the error message indicates from which system the message originated. For example, CA XCOM Data Transport message S7009 is issued by CA XCOM Data Transport for AS/400, while CA XCOM Data Transport message C1653E is issued from CA XCOM Data Transport on the PC. The various system codes are as follows:

A

Gateway

E

z/VSE

M

z/OS

N

Windows

R

Netware

S

i5/OS (AS/400)

T

HP NonStop (Tandem)

U

UNIX system or Linux

V

z/VM

8

Stratus

HP NonStop-generated Messages

HP NonStop-generated messages also aid in problem determination. These messages can be found in the log file, as shown in the following sample:

```
Entered CA XCOM Data Transport 62 on 2010/11/06, 14:19:002010/11/06, 14:19:02     CA
XCOM Data TransportT0010I
Starting CA XCOM Data Transport 6.2 Transfer on 2000/11/06, 14:19:02
2010/11/06, 14:19:03     CA XCOM Data TransportT0012I Receiving local file
'$CLX12.SCI.ptjob'
2010/11/06, 14:19:03     CA XCOM Data TransportT0016I Transfer Started
2010/11/06, 14:19:03     CA XCOM Data TransportT0002I Received 5 records, 122 bytes,
in 0 seconds (3551 bytes/sec)
2010/11/06, 14:19:03     CA XCOM Data TransportT0011I Ending Transfer
```

Calling Technical Support

Please have the following information ready before contacting CA Support:

- All the diagnostic information described in Collect Diagnostic Data
- Product name, release number, operating system and genlevel
- Product name and release number of any other software you suspect is involved
- Release level and PUTLEVEL of the operating system
- Your name, telephone number and extension (if any)
- Your company name
- Your site ID
- A severity code, a number (from 1 to 4) that you assign to the problem. Use the following to determine the severity of the problem:
    1. A "system down" or inoperative condition
    2. A suspected high-impact condition associated with the product
    3. A question concerning product performance or an intermittent low-impact condition associated with the product
    4. A question concerning general product utilization or implementation

Product Versions and Maintenance

Customers are requested to operate only under currently supported versions of CA Technologies products. Customers with current maintenance agreements also receive ongoing maintenance.

New customers can download the latest version of the product or, in most cases, order the product.

Requesting Enhancements

CA Technologies welcomes your suggestions for product enhancements. All suggestions are considered and acknowledged. To request an enhancement, you can either contact your CA Technologies account representative or visit the CA Online Support web site, http://ca.com/support.

Appendix D: ASCII/EBCDIC Translation Tables

There are two files supplied on your distribution tape that contain tables used to map ASCII-to-EBCDIC and EBCDIC-to-ASCII translation:

- ASCEBC
- EBCASC

Modify these tables to conform to national language differences.

Note: You can have multiple translation tables. Specify the appropriate table file in the ASCEBC and/or EBCASC parameters.

This section contains the following topics:

Table Reading Guidelines

Use the following guidelines to read the translation tables:

- Lines that start with a forward slash (/) are comments.
- Valid input lines contain the decimal value of the translated character starting in column 0. This decimal value is terminated with a space. Comments describing the EBCDIC and ASCII character values for this decimal value follow this space.
- ASCII-to-EBCDIC translations use the table in the ASCEBC file. The ASCII character's decimal value is the index into the table. For example, a P in ASCII is decimal 80 and hex 50. The 80th entry in the ASCEBC table is decimal 215 hex d7, which is P in EBCDIC.
- EBCDIC-to-ASCII translations use the table in the EBCASC file. The EBCDIC character's value is the index into the table. For example, a P in EBCDIC is decimal 215 and hex d7. The 215th entry in the EBCDIC table is decimal 80 and hex 50, which is P in ASCII.

The ASCII-to-EBCDIC Translation Table

The ASCII-to-EBCDIC translation table is as follows:

```
# unsigned char a2e[256]
/*               ASCII      EBCDIC       */
/*            Oct Dec 0x  Oct Dec 0x     */
0          /* 000 000 00  000 000 00 ' ' */
1          /* 001 001 01  001 001 01 ' ' */
2          /* 002 002 02  002 002 02 ' ' */
3          /* 003 003 03  003 003 03 ' ' */
55         /* 004 004 04  067 055 37 ' ' */
45         /* 005 005 05  055 045 2d ' ' */
46         /* 006 006 06  056 046 2e ' ' */
47         /* 007 007 07  057 047 2f ' ' */
22         /* 010 008 08  026 022 16 ' ' */
5          /* 011 009 09  005 005 05 ' ' */
37         /* 012 010 0a  045 037 25 ' ' */
11         /* 013 011 0b  013 011 0b ' ' */
12         /* 014 012 0c  014 012 0c ' ' */
13         /* 015 013 0d  015 013 0d ' ' */
14         /* 016 014 0e  016 014 0e ' ' */
15         /* 017 015 0f  017 015 0f ' ' */
16         /* 020 016 10  020 016 10 ' ' */
17         /* 021 017 11  021 017 11 ' ' */
18         /* 022 018 12  022 018 12 ' ' */
19         /* 023 019 13  023 019 13 ' ' */
60         /* 024 020 14  074 060 3c ' ' */
61         /* 025 021 15  075 061 3d ' ' */
50         /* 026 022 16  062 050 32 ' ' */
38         /* 027 023 17  046 038 26 ' ' */
24         /* 030 024 18  030 024 18 ' ' */
25         /* 031 025 19  031 025 19 ' ' */
63         /* 032 026 1a  077 063 3f ' ' */
39         /* 033 027 1b  047 039 27 ' ' */
28         /* 034 028 1c  034 028 1c ' ' */
29         /* 035 029 1d  035 029 1d ' ' */
30         /* 036 030 1e  036 030 1e ' ' */
31         /* 037 031 1f  037 031 1f ' ' */
64         /* 040 032 20  100 064 40 ' ' */
90         /* 041 033 21  132 090 5a '!' */
127        /* 042 034 22  177 127 7f '"' */
123        /* 043 035 23  173 123 7b '#' */
91         /* 044 036 24  133 091 5b '$' */
108        /* 045 037 25  154 108 6c '%' */
80         /* 046 038 26  120 080 50 '&' */
125        /* 047 039 27  175 125 7d '\''*/
77         /* 050 040 28  115 077 4d '(' */
93         /* 051 041 29  135 093 5d ')' */
```

```
92          /* 052 042 2a  134 092 5c '*' */
78          /* 053 043 2b  116 078 4e '+' */
107         /* 054 044 2c  153 107 6b ',' */
96          /* 055 045 2d  140 096 60 '-' */
75          /* 056 046 2e  113 075 4b '.' */
97          /* 057 047 2f  141 097 61 '/' */
240         /* 060 048 30  360 240 f0 '0' */
241         /* 061 049 31  361 241 f1 '1' */
242         /* 062 050 32  362 242 f2 '2' */
243         /* 063 051 33  363 243 f3 '3' */
244         /* 064 052 34  364 244 f4 '4' */
245         /* 065 053 35  365 245 f5 '5' */
246         /* 066 054 36  366 246 f6 '6' */
247         /* 067 055 37  367 247 f7 '7' */
248         /* 070 056 38  370 248 f8 '8' */
249         /* 071 057 39  371 249 f9 '9' */
122         /* 072 058 3a  172 122 7a ':' */
94          /* 073 059 3b  136 094 5e ';' */
76          /* 074 060 3c  114 076 4c ''  */
126         /* 075 061 3d  176 126 7e '=' */
110         /* 076 062 3e  156 110 6e ''  */
111         /* 077 063 3f  157 111 6f '?' */
124         /* 100 064 40  174 124 7c '@' */
193         /* 101 065 41  301 193 c1 'A' */
194         /* 102 066 42  302 194 c2 'B' */
195         /* 103 067 43  303 195 c3 'C' */
196         /* 104 068 44  304 196 c4 'D' */
197         /* 105 069 45  305 197 c5 'E' */
198         /* 106 070 46  306 198 c6 'F' */
199         /* 107 071 47  307 199 c7 'G' */
200         /* 110 072 48  310 200 c8 'H' */
201         /* 111 073 49  311 201 c9 'I' */
209         /* 112 074 4a  321 209 d1 'J' */
210         /* 113 075 4b  322 210 d2 'K' */
211         /* 114 076 4c  323 211 d3 'L' */
212         /* 115 077 4d  324 212 d4 'M' */
213         /* 116 078 4e  325 213 d5 'N' */
214         /* 117 079 4f  326 214 d6 'O' */
215         /* 120 080 50  327 215 d7 'P' */
216         /* 121 081 51  330 216 d8 'Q' */
217         /* 122 082 52  331 217 d9 'R' */
226         /* 123 083 53  342 226 e2 'S' */
227         /* 124 084 54  343 227 e3 'T' */
228         /* 125 085 55  344 228 e4 'U' */
229         /* 126 086 56  345 229 e5 'V' */
230         /* 127 087 57  346 230 e6 'W' */
231         /* 130 088 58  347 231 e7 'X' */
232         /* 131 089 59  350 232 e8 'Y' */
233         /* 132 090 5a  351 233 e9 'Z' */
```

```
173        /* 133 091 5b  255 173 ad '[' */
224        /* 134 092 5c  340 224 e0 '\' */
189        /* 135 093 5d  275 189 bd ']' */
95         /* 136 094 5e  137 095 5f '^' */
109        /* 137 095 5f  155 109 6d '_' */
121        /* 140 096 60  171 121 79 ''  */
129        /* 141 097 61  201 129 81 'a' */
130        /* 142 098 62  202 130 82 'b' */
131        /* 143 099 63  203 131 83 'c' */
132        /* 144 100 64  204 132 84 'd' */
133        /* 145 101 65  205 133 85 'e' */
134        /* 146 102 66  206 134 86 'f' */
135        /* 147 103 67  207 135 87 'g' */
136        /* 150 104 68  210 136 88 'h' */
137        /* 151 105 69  211 137 89 'i' */
145        /* 152 106 6a  221 145 91 'j' */
146        /* 153 107 6b  222 146 92 'k' */
147        /* 154 108 6c  223 147 93 'l' */
148        /* 155 109 6d  224 148 94 'm' */
149        /* 156 110 6e  225 149 95 'n' */
150        /* 157 111 6f  226 150 96 'o' */
151        /* 160 112 70  227 151 97 'p' */
152        /* 161 113 71  230 152 98 'q' */
153        /* 162 114 72  231 153 99 'r' */
162        /* 163 115 73  242 162 a2 's' */
163        /* 164 116 74  243 163 a3 't' */
164        /* 165 117 75  244 164 a4 'u' */
165        /* 166 118 76  245 165 a5 'v' */
166        /* 167 119 77  246 166 a6 'w' */
167        /* 170 120 78  247 167 a7 'x' */
168        /* 171 121 79  250 168 a8 'y' */
169        /* 172 122 7a  251 169 a9 'z' */
192        /* 173 123 7b  300 192 c0 '{' */
79         /* 174 124 7c  117 079 4f '|' */
208        /* 175 125 7d  320 208 d0 '}' */
# 161      /* 176 126 7e  241 161 a1 '~' */
95         /* 176 126 7e  241 161 a1 '~' */
7          /* 177 127 7f  007 007 07 ' ' */
0          /* 200 128 80  000 000 00 ' ' */
0          /* 201 129 81  001 001 01 ' ' */
0          /* 202 130 82  002 002 02 ' ' */
0          /* 203 131 83  003 003 03 ' ' */
0          /* 204 132 84  067 055 37 ' ' */
0          /* 205 133 85  055 045 2d ' ' */
0          /* 206 134 86  056 046 2e ' ' */
0          /* 207 135 87  057 047 2f ' ' */
0          /* 210 136 88  026 022 16 ' ' */
0          /* 211 137 89  005 005 05 ' ' */
0          /* 212 138 8a  045 037 25 ' ' */
```

```
0          /* 213 139 8b  013 011 0b ' ' */
0          /* 214 140 8c  014 012 0c ' ' */
0          /* 215 141 8d  015 013 0d ' ' */
0          /* 216 142 8e  016 014 0e ' ' */
0          /* 217 143 8f  017 015 0f ' ' */
0          /* 220 144 90  020 016 10 ' ' */
0          /* 221 145 91  021 017 11 ' ' */
0          /* 222 146 92  022 018 12 ' ' */
0          /* 223 147 93  023 019 13 ' ' */
0          /* 224 148 94  074 060 3c ' ' */
0          /* 225 149 95  075 061 3d ' ' */
0          /* 226 150 96  062 050 32 ' ' */
0          /* 227 151 97  046 038 26 ' ' */
0          /* 230 152 98  030 024 18 ' ' */
0          /* 231 153 99  031 025 19 ' ' */
0          /* 232 154 9a  077 063 3f ' ' */
0          /* 233 155 9b  047 039 27 ' ' */
0          /* 234 156 9c  034 028 1c ' ' */
0          /* 235 157 9d  035 029 1d ' ' */
0          /* 236 158 9e  036 030 1e ' ' */
0          /* 237 159 9f  037 031 1f ' ' */
0          /* 240 160 a0  100 064 40 ' ' */
0          /* 241 161 a1  132 090 5a ' ' */
0          /* 242 162 a2  177 127 7f ' ' */
0          /* 243 163 a3  173 123 7b ' ' */
0          /* 244 164 a4  133 091 5b ' ' */
0          /* 245 165 a5  154 108 6c ' ' */
161        /* 246 166 a6  120 080 50 '&' */
0          /* 247 167 a7  175 125 7d ' '' */
0          /* 250 168 a8  115 077 4d ' ' */
0          /* 251 169 a9  135 093 5d ' ' */
0          /* 252 170 aa  134 092 5c ' ' */
0          /* 253 171 ab  116 078 4e ' ' */
0          /* 254 172 ac  153 107 6b ' ' */
0          /* 255 173 ad  140 096 60 ' ' */
0          /* 256 174 ae  113 075 4b ' ' */
0          /* 257 175 af  141 097 61 ' ' */
0          /* 260 176 b0  360 240 f0 ' ' */
0          /* 261 177 b1  361 241 f1 ' ' */
0          /* 262 178 b2  362 242 f2 ' ' */
0          /* 263 179 b3  363 243 f3 ' ' */
0          /* 264 180 b4  364 244 f4 ' ' */
0          /* 265 181 b5  365 245 f5 ' ' */
0          /* 266 182 b6  366 246 f6 ' ' */
0          /* 267 183 b7  367 247 f7 ' ' */
0          /* 270 184 b8  370 248 f8 ' ' */
0          /* 271 185 b9  371 249 f9 ' ' */
0          /* 272 186 ba  172 122 7a ' ' */
0          /* 273 187 bb  136 094 5e ' ' */
```

```
0          /* 274 188 bc  114 076 4c ' ' */
0          /* 275 189 bd  176 126 7e ' ' */
0          /* 276 190 be  156 110 6e ' ' */
0          /* 277 191 bf  157 111 6f ' ' */
0          /* 300 192 c0  174 124 7c ' ' */
30         /* 301 193 c1  301 193  1 ' ' */
28         /* 302 194 c2  302 194 c2 ' ' */
0          /* 303 195 c3  303 195 c3 ' ' */
0          /* 304 196 c4  304 196 c4 ' ' */
0          /* 305 197 c5  305 197 c5 ' ' */
0          /* 306 198 c6  306 198 c6 ' ' */
0          /* 307 199 c7  307 199 c7 ' ' */
0          /* 310 200 c8  310 200 c8 ' ' */
0          /* 311 201 c9  311 201 c9 ' ' */
0          /* 312 202 ca  321 209 d1 ' ' */
0          /* 313 203 cb  322 210 d2 ' ' */
0          /* 314 204 cc  323 211 d3 ' ' */
0          /* 315 205 cd  324 212 d4 ' ' */
0          /* 316 206 ce  325 213 d5 ' ' */
0          /* 317 207 cf  326 214 d6 ' ' */
0          /* 320 208 d0  327 215 d7 ' ' */
123        /* 321 209 d1  330 216 d8 ' ' */
0          /* 322 210 d2  331 217 d9 ' ' */
0          /* 323 211 d3  342 226 e2 ' ' */
0          /* 324 212 d4  343 227 e3 ' ' */
0          /* 325 213 d5  344 228 e4 ' ' */
0          /* 326 214 d6  345 229 e5 ' ' */
0          /* 327 215 d7  346 230 e6 ' ' */
0          /* 330 216 d8  347 231 e7 ' ' */
0          /* 331 217 d9  350 232 e8 ' ' */
0          /* 332 218 da  351 233 e9 ' ' */
0          /* 333 219 db  255 173 ad ' ' */
0          /* 334 220 dc  340 224 e0 ' ' */
0          /* 335 221 dd  275 189 bd ' ' */
0          /* 336 222 de  137 095 5f ' ' */
0          /* 337 223 df  155 109 6d ' ' */
0          /* 340 224 e0  171 121 79 ' ' */
0          /* 341 225 e1  201 129 81 ' ' */
0          /* 342 226 e2  202 130 82 ' ' */
0          /* 343 227 e3  203 131 83 ' ' */
0          /* 344 228 e4  204 132 84 ' ' */
0          /* 345 229 e5  205 133 85 ' ' */
0          /* 346 230 e6  206 134 86 ' ' */
0          /* 347 231 e7  207 135 87 ' ' */
0          /* 350 232 e8  210 136 88 ' ' */
0          /* 351 233 e9  211 137 89 ' ' */
0          /* 352 234 ea  221 145 91 ' ' */
0          /* 353 235 eb  222 146 92 ' ' */
0          /* 354 236 ec  223 147 93 ' ' */
```

```
0          /* 355 237 ed  224 148 94 ' ' */
0          /* 356 238 ee  225 149 95 ' ' */
0          /* 357 239 ef  226 150 96 ' ' */
0          /* 360 240 f0  227 151 97 ' ' */
106        /* 361 241 f1  230 152 98 ' ' */
0          /* 362 242 f2  231 153 99 ' ' */
0          /* 363 243 f3  242 162 a2 ' ' */
0          /* 364 244 f4  243 163 a3 ' ' */
0          /* 365 245 f5  244 164 a4 ' ' */
0          /* 366 246 f6  245 165 a5 ' ' */
0          /* 367 247 f7  246 166 a6 ' ' */
0          /* 370 248 f8  247 167 a7 ' ' */
0          /* 371 249 f9  250 168 a8 ' ' */
0          /* 372 250 fa  251 169 a9 ' ' */
0          /* 373 251 fb  300 192 c0 ' ' */
0          /* 374 252 fc  117 079 4f ' ' */
0          /* 375 253 fd  320 208 d0 ' ' */
0          /* 376 254 fe  241 161 a1 ' ' */
0          /* 377 255 ff  007 007 07 ' ' */
```

The EBCDIC-to-ASCII Translation Table

The EBCDIC-to-ASCII translation table is as follows:

```
/          ebcdic      ascii         **/
/
/          oct dec hex oct dec hex     /
/
0       /* 000 000 00  000 000 00 ' ' */
1       /* 001 001 01  001 001 01 ' ' */
2       /* 002 002 02  002 002 02 ' ' */
3       /* 003 003 03  003 003 03 ' ' */
28      /* 004 004 04  034 028 1c ' ' */
9       /* 005 005 05  011 009 09 ' ' */
6       /* 006 006 06  006 006 06 ' ' */
127     /* 007 007 07  177 127 7f ' ' */
23      /* 010 008 08  027 023 17 ' ' */
13      /* 011 009 09  015 013 0d ' ' */
14      /* 012 010 0a  016 014 0e ' ' */
11      /* 013 011 0b  013 011 0b ' ' */
12      /* 014 012 0c  014 012 0c ' ' */
13      /* 015 013 0d  015 013 0d ' ' */
14      /* 016 014 0e  016 014 0e ' ' */
15      /* 017 015 0f  017 015 0f ' ' */
16      /* 020 016 10  020 016 10 ' ' */
17      /* 021 017 11  021 017 11 ' ' */
18      /* 022 018 12  022 018 12 ' ' */
19      /* 023 019 13  023 019 13 ' ' */
29      /* 024 020 14  035 029 1d ' ' */
5       /* 025 021 15  005 005 05 ' ' */
8       /* 026 022 16  010 008 08 ' ' */
7       /* 027 023 17  007 007 07 ' ' */
24      /* 030 024 18  030 024 18 ' ' */
25      /* 031 025 19  031 025 19 ' ' */
18      /* 032 026 1a  022 018 12 ' ' */
15      /* 033 027 1b  017 015 0f ' ' */
28      /* 034 028 1c  034 028 1c ' ' */
29      /* 035 029 1d  035 029 1d ' ' */
30      /* 036 030 1e  036 030 1e ' ' */
31      /* 037 031 1f  037 031 1f ' ' */
0       /* 040 032 20  000 000 00 ' ' */
1       /* 041 033 21  001 001 01 ' ' */
2       /* 042 034 22  002 002 02 ' ' */
3       /* 043 035 23  003 003 03 ' ' */
4       /* 044 036 24  004 004 04 ' ' */
10      /* 045 037 25  012 010 0a ' ' */
23      /* 046 038 26  027 023 17 ' ' */
27      /* 047 039 27  033 027 1b ' ' */
8       /* 050 040 28  010 008 08 ' ' */
```

```
9       /* 051 041 29  011 009 09 ' ' */
10      /* 052 042 2a  012 010 0a ' ' */
11      /* 053 043 2b  013 011 0b ' ' */
12      /* 054 044 2c  014 012 0c ' ' */
5       /* 055 045 2d  005 005 05 ' ' */
6       /* 056 046 2e  006 006 06 ' ' */
7       /* 057 047 2f  007 007 07 ' ' */
16      /* 060 048 30  020 016 10 ' ' */
17      /* 061 049 31  021 017 11 ' ' */
22      /* 062 050 32  026 022 16 ' ' */
19      /* 063 051 33  023 019 13 ' ' */
20      /* 064 052 34  024 020 14 ' ' */
21      /* 065 053 35  025 021 15 ' ' */
22      /* 066 054 36  026 022 16 ' ' */
4       /* 067 055 37  004 004 04 ' ' */
24      /* 070 056 38  030 024 18 ' ' */
25      /* 071 057 39  031 025 19 ' ' */
26      /* 072 058 3a  032 026 1a ' ' */
27      /* 073 059 3b  033 027 1b ' ' */
20      /* 074 060 3c  024 020 14 ' ' */
21      /* 075 061 3d  025 021 15 ' ' */
30      /* 076 062 3e  036 030 1e ' ' */
26      /* 077 063 3f  032 026 1a ' ' */
32      /* 100 064 40  040 032 20 ' ' */
32      /* 101 065 41  040 032 20 ' ' */
33      /* 102 066 42  041 033 21 '!' */
34      /* 103 067 43  042 034 22 '"' */
35      /* 104 068 44  043 035 23 '#' */
36      /* 105 069 45  044 036 24 '$' */
37      /* 106 070 46  045 037 25 '%' */
38      /* 107 071 47  046 038 26 '&' */
39      /* 110 072 48  047 039 27 '\'' */
40      /* 111 073 49  050 040 28 '(' */
85      /* 112 074 4a  125 085 55 'U' */
46      /* 113 075 4b  056 046 2e '.' */
60      /* 114 076 4c  074 060 3c '' */
40      /* 115 077 4d  050 040 28 '(' */
43      /* 116 078 4e  053 043 2b '+' */
124     /* 117 079 4f  174 124 7c '|' */
38      /* 120 080 50  046 038 26 '&' */
41      /* 121 081 51  051 041 29 ')' */
42      /* 122 082 52  052 042 2a '*' */
43      /* 123 083 53  053 043 2b '+' */
44      /* 124 084 54  054 044 2c ',' */
45      /* 125 085 55  055 045 2d '-' */
46      /* 126 086 56  056 046 2e '.' */
47      /* 127 087 57  057 047 2f '/' */
48      /* 130 088 58  060 048 30 '0' */
49      /* 131 089 59  061 049 31 '1' */
```

```
33      /* 132 090 5a  041 033 21 '!' */
36      /* 133 091 5b  044 036 24 '$' */
42      /* 134 092 5c  052 042 2a '*' */
41      /* 135 093 5d  051 041 29 ')' */
59      /* 136 094 5e  073 059 3b ';' */
126     /* 137 095 5f  176 126 7e '~' */
45      /* 140 096 60  055 045 2d '-' */
47      /* 141 097 61  057 047 2f '/' */
50      /* 142 098 62  062 050 32 '2' */
51      /* 143 099 63  063 051 33 '3' */
52      /* 144 100 64  064 052 34 '4' */
53      /* 145 101 65  065 053 35 '5' */
54      /* 146 102 66  066 054 36 '6' */
55      /* 147 103 67  067 055 37 '7' */
56      /* 150 104 68  070 056 38 '8' */
57      /* 151 105 69  071 057 39 '9' */
241     /* 152 106 6a  113 075 4b ' ' */
44      /* 153 107 6b  054 044 2c ',' */
37      /* 154 108 6c  045 037 25 '%' */
95      /* 155 109 6d  137 095 5f '_' */
62      /* 156 110 6e  076 062 3e '' */
63      /* 157 111 6f  077 063 3f '?' */
58      /* 160 112 70  072 058 3a ':' */
59      /* 161 113 71  073 059 3b ';' */
60      /* 162 114 72  074 060 3c '' */
61      /* 163 115 73  075 061 3d '=' */
62      /* 164 116 74  076 062 3e '' */
63      /* 165 117 75  077 063 3f '?' */
64      /* 166 118 76  100 064 40 '@' */
65      /* 167 119 77  101 065 41 'A' */
66      /* 170 120 78  102 066 42 'B' */
96      /* 171 121 79  140 096 60 '' */
58      /* 172 122 7a  072 058 3a ':' */
35      /* 173 123 7b  043 035 23 ' ' */
64      /* 174 124 7c  100 064 40 '@' */
39      /* 175 125 7d  047 039 27 '\'' */
61      /* 176 126 7e  075 061 3d '=' */
34      /* 177 127 7f  042 034 22 '"' */
67      /* 200 128 80  103 067 43 'C' */
97      /* 201 129 81  141 097 61 'a' */
98      /* 202 130 82  142 098 62 'b' */
99      /* 203 131 83  143 099 63 'c' */
100     /* 204 132 84  144 100 64 'd' */
101     /* 205 133 85  145 101 65 'e' */
102     /* 206 134 86  146 102 66 'f' */
103     /* 207 135 87  147 103 67 'g' */
104     /* 210 136 88  150 104 68 'h' */
105     /* 211 137 89  151 105 69 'i' */
68      /* 212 138 8a  104 068 44 'D' */
```

```
69      /* 213 139 8b  105 069 45 'E' */
70      /* 214 140 8c  106 070 46 'F' */
71      /* 215 141 8d  107 071 47 'G' */
72      /* 216 142 8e  110 072 48 'H' */
73      /* 217 143 8f  111 073 49 'I' */
74      /* 220 144 90  112 074 4a 'J' */
106     /* 221 145 91  152 106 6a 'j' */
107     /* 222 146 92  153 107 6b 'k' */
108     /* 223 147 93  154 108 6c 'l' */
109     /* 224 148 94  155 109 6d 'm' */
110     /* 225 149 95  156 110 6e 'n' */
111     /* 226 150 96  157 111 6f 'o' */
112     /* 227 151 97  160 112 70 'p' */
113     /* 230 152 98  161 113 71 'q' */
114     /* 231 153 99  162 114 72 'r' */
94      /* 232 154 9a  136 094 5e '~' */
76      /* 233 155 9b  114 076 4c 'L' */
77      /* 234 156 9c  115 077 4d 'M' */
78      /* 235 157 9d  116 078 4e 'N' */
79      /* 236 158 9e  117 079 4f 'O' */
80      /* 237 159 9f  120 080 50 'P' */
81      /* 240 160 a0  121 081 51 'Q' */
166     /* 241 161 a1  145 101 65 'e' */
115     /* 242 162 a2  163 115 73 's' */
116     /* 243 163 a3  164 116 74 't' */
117     /* 244 164 a4  165 117 75 'u' */
118     /* 245 165 a5  166 118 76 'v' */
119     /* 246 166 a6  167 119 77 'w' */
120     /* 247 167 a7  170 120 78 'x' */
121     /* 250 168 a8  171 121 79 'y' */
122     /* 251 169 a9  172 122 7a 'z' */
82      /* 252 170 aa  122 082 52 'R' */
83      /* 253 171 ab  123 083 53 'S' */
84      /* 254 172 ac  124 084 54 'T' */
91      /* 255 173 ad  133 091 5b '[' */
86      /* 256 174 ae  126 086 56 'V' */
87      /* 257 175 af  127 087 57 'W' */
88      /* 260 176 b0  130 088 58 'X' */
89      /* 261 177 b1  131 089 59 'Y' */
90      /* 262 178 b2  132 090 5a 'Z' */
91      /* 263 179 b3  133 091 5b '[' */
92      /* 264 180 b4  134 092 5c '\' */
93      /* 265 181 b5  135 093 5d ']' */
94      /* 266 182 b6  136 094 5e '~' */
95      /* 267 183 b7  137 095 5f '_' */
96      /* 270 184 b8  140 096 60 '' */
97      /* 271 185 b9  141 097 61 'a' */
98      /* 272 186 ba  142 098 62 'b' */
99      /* 273 187 bb  143 099 63 'c' */
```

```
100     /* 274 188 bc  144 100 64 'd' */
93      /* 275 189 bd  135 093 5d ']' */
102     /* 276 190 be  146 102 66 'f' */
103     /* 277 191 bf  147 103 67 'g' */
123     /* 300 192 c0  173 123 7b '{' */
65      /* 301 193 c1  101 065 41 'A' */
66      /* 302 194 c2  102 066 42 'B' */
67      /* 303 195 c3  103 067 43 'C' */
68      /* 304 196 c4  104 068 44 'D' */
69      /* 305 197 c5  105 069 45 'E' */
70      /* 306 198 c6  106 070 46 'F' */
71      /* 307 199 c7  107 071 47 'G' */
72      /* 310 200 c8  110 072 48 'H' */
73      /* 311 201 c9  111 073 49 'I' */
104     /* 312 202 ca  150 104 68 'h' */
105     /* 313 203 cb  151 105 69 'i' */
106     /* 314 204 cc  152 106 6a 'j' */
107     /* 315 205 cd  153 107 6b 'k' */
108     /* 316 206 ce  154 108 6c 'l' */
109     /* 317 207 cf  155 109 6d 'm' */
125     /* 320 208 d0  175 125 7d '}' */
74      /* 321 209 d1  112 074 4a 'J' */
75      /* 322 210 d2  113 075 4b 'K' */
76      /* 323 211 d3  114 076 4c 'L' */
77      /* 324 212 d4  115 077 4d 'M' */
78      /* 325 213 d5  116 078 4e 'N' */
79      /* 326 214 d6  117 079 4f 'O' */
80      /* 327 215 d7  120 080 50 'P' */
81      /* 330 216 d8  121 081 51 'Q' */
82      /* 331 217 d9  122 082 52 'R' */
110     /* 332 218 da  156 110 6e 'n' */
111     /* 333 219 db  157 111 6f 'o' */
112     /* 334 220 dc  160 112 70 'p' */
113     /* 335 221 dd  161 113 71 'q' */
114     /* 336 222 de  162 114 72 'r' */
115     /* 337 223 df  163 115 73 's' */
92      /* 340 224 e0  134 092 5c '\' */
31      /* 341 225 e1  037 031 1f ' ' */
83      /* 342 226 e2  123 083 53 'S' */
84      /* 343 227 e3  124 084 54 'T' */
85      /* 344 228 e4  125 085 55 'U' */
86      /* 345 229 e5  126 086 56 'V' */
87      /* 346 230 e6  127 087 57 'W' */
88      /* 347 231 e7  130 088 58 'X' */
89      /* 350 232 e8  131 089 59 'Y' */
90      /* 351 233 e9  132 090 5a 'Z' */
116     /* 352 234 ea  164 116 74 't' */
117     /* 353 235 eb  165 117 75 'u' */
118     /* 354 236 ec  166 118 76 'v' */
```

```
119     /* 355 237 ed  167 119 77 'w' */
120     /* 356 238 ee  170 120 78 'x' */
121     /* 357 239 ef  171 121 79 'y' */
48      /* 360 240 f0  060 048 30 '0' */
49      /* 361 241 f1  061 049 31 '1' */
50      /* 362 242 f2  062 050 32 '2' */
51      /* 363 243 f3  063 051 33 '3' */
52      /* 364 244 f4  064 052 34 '4' */
53      /* 365 245 f5  065 053 35 '5' */
54      /* 366 246 f6  066 054 36 '6' */
55      /* 367 247 f7  067 055 37 '7' */
56      /* 370 248 f8  070 056 38 '8' */
57      /* 371 249 f9  071 057 39 '9' */
122     /* 372 250 fa  172 122 7a 'z' */
123     /* 373 251 fb  173 123 7b '{' */
124     /* 374 252 fc  174 124 7c '|' */
125     /* 375 253 fd  175 125 7d '}' */
126     /* 376 254 fe  176 126 7e '~' */
127     /* 377 255 ff  177 127 7f ' ' */
```

Appendix E: About Logical Units

This appendix explains the various logical unit (LU) types and discusses independent logical units (ILUs) and other pertinent issues.

This section contains the following topics:

Parts of an SNA Network

This section describes the parts of an SNA network.

LU Connections

A Logical Unit (LU) is the addressable connection point into an SNA network through which an end-user can send and receive messages. An LU is a set of rules and responsibilities. LUs can be either dependent or independent, and each LU type is associated with a protocol (for example, LU 0, LU 6.2).

The LU provides a connection into SNA for the end-user, which may either be an individual or a transaction program (for example, CA XCOM Data Transport). It allows end-users to communicate with each other and with other network addressable units (NAUs) in the network.

Logical and Physical Network Components

An SNA network is divided into physical and logical components.

The physical network consists of the following:

- Actual processors called nodes
- Data links between the nodes

The logical network consists of a set of software components called NAUs that include the following:

- Logical units (LUs)
- Physical units (PUs)
- System services control points (SSCPs)

Sessions

A session is a logical connection between two NAUs. Although several types of sessions exist, the end-user is aware of only one type that is LU-to-LU. Sessions are established when one LU sends another LU an SNA request known as a BIND. Each session has its own procedure correlation identifier (PCID).

PCIDs

A PCID is an eight-byte field placed in the BIND, UNBIND, and other SNA requests to help an LU distinguish one session from another. It is required when you are running parallel sessions.

A PCID is also known as a session identifier (SID) in VTAM displays. For each session, VTAM prompts you to note the primary or secondary node and displays the Session ID (SID) in hex. This SID is the PCID. If a trace of the BIND is taken, the PCID vector is towards the end.

The following VTAM operator command lists all sessions generated for that LU:

```
D NET,ID=<luname>,E
```

LUs

This section describes the different LU types.

IBM Strategic LU

LU 6.2 is the only LU type that is crucial to IBM long-term strategy. The 3270 data stream will be moved on top of LU 6.2, and none of the other LU types are strategic (for example, they are not considered in IBM's long-term plans).

LU Types

IBM classifies LUs into roughly seven different types (LU 6.2 is a subset of LU 6). The products that support each of these LU types will continue to be supported in future years and the 3270 data stream will also be preserved, but not in its current form.

The following list shows all of the LU types that are in use today to differing degrees.

0

Denotes a flexible protocol, which eliminates standardization beyond layers of SNA. This was commonly used in the late 1970s (before the advent of LU 6.2).

1

Specifies the protocol used as early as the 1960s by remote job entry (RJE) devices such as the 3770 RJE terminal. Designed for use with printers and card readers, this protocol is most typically used in asymmetrical links where one node is a slave to the host.

2

Specifies the protocol for 3270 video display stations. It defines the data streams used by dumb terminals to communicate with the host.

3

Specifies a variant subset of 3270 protocol that was used to drive printers attached to 3274 cluster controllers. Today it is still used to support old hardware.

4

Specifies a protocol that was intended for use on word processors attached to a host network. You can still see it on old IBM word processors.

6.1

Specifies SNA's prototype protocol defined for program-to-program communication that was developed during the late 1970s. It was a first attempt to provide a standardized mechanism for communication between intelligent peer systems.

6.2

Alternatively referred to by the marketing title Advanced Program-to-Program Communications (APPC), this used to be called the Convergent LU, or the LU type around which the entire IBM product line would converge. LU 6.2 defines standard functions or verbs such as SEND, RECEIVE, and CONFIRM that simplify the work of making two different programs on two different kinds of system talk to each other.

7

Specifies the data stream of the 5250 video display stations commonly used with the IBM midrange systems.

ILUs

CA XCOM Data Transport supports ILUs. An ILU is a logical unit that can generate sessions independent of the host. An ILU also meets the following criteria:

- It utilizes LU 6.2.
- It works on top of PU 2.1.
- It functions as a primary logical unit and therefore can send a BIND.
- It supports an extended BIND (one that contains a PCID) and works with the Network Control Program (NCP) PU 2.1 support.

Systems that currently support ILUs include the following:

- i5/OS(AS/400)
- z/OS
- z/VSE
- MS Windows
- VM (all versions)
- UNIX or Linux
- Netware

LU 6.2 Independent Implementations

Only Type 6 logical units can be independent. All other LU types are dependent. However, not all LU 6.2 implementations are independent.

Not every LU 6.2/PU 2.1 implementation can work with independent LUs. There are some aspects of PU 2.1 that NCP requires with which not all PU 2.1 implementations will work correctly. This reflects the fact that not all midrange and PC SNA Gateway vendors had the latest NCP and VTAM for testing.

PU 2.1 support can be enhanced to work with ILUs without changes to CA XCOM Data Transport. NCP supports ILUs over SDLC, the most common configuration using ILUs. A local area network gateway attached through an SDLC link to a host can also use ILUs. NCP also supports ILUs over a token ring through the TIC.

Direct Sessions with Dependent Logical Unit

An independent logical unit can have an LU 6.2 session with a dependent LU. This allows for direct sessions from an i5/OS(AS/400) to a OpenVMS over the SNA background network, even though the VTAM is PU Type 2.0. In this environment, the VTAM LOGAPPL parameter and the VTAM VARY NET LOGON command will not work.

Note: The ILU must initiate the session; it must send to the BIND.

PU Type

When using ILUs with VTAM and Netview displays, VTAM shows the PU type in its status display (PU Type 2 or PU Type 2.1). All PUs originally appears as PU 2.0. Once they become active, they display as PU 2.1.

Glossary

ABEND

Certain errors can cause an abnormal ending (ABEND) of the transaction program. When the LU terminates a program because of an ABEND condition, it deallocates all active conversations.

Access Code

A code used in access control lists to show access rights.

Access Control

An operating system mechanism which determines a user's access rights to files and directories.

Access Mode

The method the I/O system uses to access records for reading or writing. The access modes are sequential, random, and indexed.

ACF

See Advanced Communications Function.

Adjacent Node

A node that is directly connected (either logically or physically) to another node. A pair of adjacent SNA Type 2.1 Nodes can be directly connected in a peer-to-peer manner. This is referred to as a Low Entry Networking (LEN) connection.

Advanced Communications Function (ACF)

A prefix given to IBM's mainframe communications packages to differentiate between the old releases of VTAM, NCP, and SSP, which came with the MVS operating system.

Advanced Peer-to-Peer Networking (APPN)

An IBM technology that allows networking of multiple Type 2.1 Nodes without requiring a System/370 host. APPN nodes provide intermediate node network routing and support dynamic networking facilities, such as network reconfiguration, locating users in the network. APPN software is implemented on the System/36 and AS/400.

Advanced Program-to-Program Communications (APPC)

The marketing term for SNA LU 6.2 program-to-program communications. APPC/LU 6.2 provides a generalized communications vehicle that can be used by transaction programs to exchange information. IBM transaction programs such as DIS, SNA/DS, and DDM use LU 6.2 for their program-to-program communications.

American National Standards Institute (ANSI)

This is a United States organization that was created by the computer industry to establish computing standards. Participation in these standards is voluntary. (See also American Standard Code for Information Interchange.)

American Standard Code for Information Interchange (ASCII)

The standard binary code prepared and recommended by the American National Standards Institute to represent characters in a computer system. This system is used by most non-IBM computers. With ASCII, each character is represented in a byte consisting of 7 bits of data and 1 parity bit. (See also Extended Binary-Coded Decimal Interchange Code.)

API

See Application Program Interface.

APPC

See Advanced Program-to-Program Communications.

Application

A program or set of programs which uses a data communication network in order to provide remote users with computer services or information. In SNA networks, application programs are considered end-users of the network and are represented by Logical Units (LUs).

Application Program Interface (API)

A set of calls that application programs use which access services provided by supporting software. For example, LU 6.2 Transaction Programs interface to the supporting LU via the LU's application program interface.

APPN

See Advanced Peer-to-Peer Networking.

Architecture

A set of rules and definitions describing the structure and operation of a network. Systems Network Architecture (SNA) is IBM's set of rules and definitions describing how their systems will interconnect and operate in a network.

AS/400

IBM's strategic, multi-user, midrange applications processing system. The AS/400 supports a wide range of SNA communications, including 3270 emulation and peer-to-peer communications using Logical Unit Type 6.2. The AS/400 is the successor to IBM's System/36 and System/38 product lines.

ASCII

See American Standard Code for Information Interchange.

AST

See Asynchronous System Trap.

Asynchronous

A method of data transmission where every character sent is bounded by a start and stop bit, and where no timing or clocking information is exchanged between parties.

Asynchronous Communications

A communications protocol standard for data transmission that does not allow for the buffering of data or polling. The flow of data is maintained by start and stop control bits, not timing sequences. Thus, the timing interval between the transmission of each character can vary from one character to the next. This protocol is not recognized by IBM's SNA suite of protocols. See also Binary Synchronous Communications and Synchronous Data Link Control.

Asynchronous System Trap (AST)

A VAX specific term. ASTs are a mechanism for signalling asynchronous events to a process. Specifically, a procedure (or routine) designated by either the process or the system executes in the context of the process. An example of these services is documented in the *VAX/VMS System Services Reference Manual.*

Automatic Dialer

A device that generates pulse or touch tone dialing sequences. An automatic dialer is used by computers and terminals to directly dial a call. Many modems have automatic dial features.

Basic Conversations

Conversations that require the transaction to build and interpret generalized data stream (GDS) headers. The transaction is responsible for error recovery and data stream mapping.

Batch Process

A non-interactive process that executes a command or command macro. When you request a batch process, from the System/88 or Stratus Operating System for instance, the batch request is put into a specified queue and a batch process starts to execute the command whenever resources become available. The batch process runs independently of the process that issued the batch request.

Batch Task

A collection of related functions that run sequentially, usually without interaction with a user.

Baud

A unit of signal variation rate on a communication line. In modern usage, it is normally (and incorrectly) used in place of "bits per second." In data communications, each baud can encode one or more binary bits of computer data. Typically, 1, 2, or 4 bits are encoded.

Binary Synchronous Communications

A data communications protocol that allows for buffering and polling. Although supported by some IBM hardware and software, it is not a recognized protocol in the SNA suite of protocols. See also Asynchronous Communications and Synchronous Data Link Control.

Bisync

See Binary Synchronous Communication.

Bit

One binary digit of a computer's representation of information. There are normally eight bits in a computer character (byte).

Block

A set of information sent on a communication line. Normally associated with synchronous protocols and sometimes called a frame, a disk block holds 4096 bytes. It is the smallest addressable unit in disk secondary storage and is the unit of transfer of data between main storage and secondary storage. File and directory blocks are read into the buffer pool before being accessed by the system.

Block I/O

A method of transferring data between main and secondary storage that moves a block (4096 bytes) at a time, disregarding the file and record structures of the data.

Buffer

A memory area reserved for use in performing input/output (I/O) operations.

Byte

A collection of 8 bits which make up a letter, number, symbol, and so on. Normally used as another term for character, byte is an IBM term. An unsigned byte variable can contain integer values in the range of 0 to 255; a signed byte variable can contain integer values in the range of -128 to 128.

CA Roscoe

A software package from CA that is a partial alternative to IBM's TSO; CA Roscoe is less powerful than TSO, but it imposes significantly less overhead.

Carrier

Either a company that provides communication circuits to the public or a signal onto which digital information is placed for transmission. In popular modern usage, "I have carrier" means that the signal from the remote modem is reaching the local modem requirement for communication.

CCITT

International Consultative Committee for Telephone and Telegraph, a United Nations affiliate which proposes worldwide public standards for computer networking protocols. The International Standards Organization (ISO) usually shows agreement with CCITT proposals when it announces new protocol standards every four years.

CCS

See Common Communications Support.

Channel

A path over which information can flow between two points, sometimes used interchangeably with terms such as circuit, line, trunk, or path.

CICS

See Customer Information Control System.

Cluster Controller

An SNA term used to describe Peripheral Nodes which control multiple, clustered devices such as display stations and printers. This term is commonly used when talking about 3270 control units. A cluster controller node is now referred to as a Peripheral Node.

Command Macro

A user-written program, invoked from the command line. A command macro is composed of calls to the operating system commands, calls to user programs and commands, and command macro statements. The command processor reads and executes lines from the macro file until it either reaches the end of the file or reaches the terminating macro statement.

Common Communications Support (CCS)

An SAA element that consists of components used for communications between SAA systems. The CCS element is subdivided into categories of components including Objects, Data Streams, Application Services, Session Services, Network, and Data Link Controls. The intent of this element is to allow inter-operability between SAA systems.

Common Programming Interface (CPI)

An SAA element that is subdivided into categories of programming languages and services components. This element is used to standardize the available programming environments on SAA systems. SAA applications are developed using CPI components and interfaces.

Common Programming Interface for Communications (CPIC)

The LU 6.2-based application programming interface defined as part of SAA's CPI element. The CPIC interface is used by application programs for communicating with other application programs.

Common User Access (CUA)

An SAA element that contains specifications and guidelines for an application's user interface. This element is used to standardize the user interface provided by SAA application programs. CUA includes support for both non-programmable terminals, such as 3270s, and intelligent workstations, such as PS/2s running OS/2.

Communication Controller Node

An SNA network node which acts as a "pure" communications processor to handle functions like network routing and control of physical data links. IBM hardware products like the 3720, 3725, and 3745 act as SNA communications controllers when they are running the Network Control Program (NCP) software. SNA communications controllers are also referred to as Type 4 Nodes.

Configuration Table

One of the table files that the operating system uses to identify the elements of a system.

Control Point (CP)

The intelligence that manages a node and provides network services to attached logical units.

Conversation

The name for communications between two programs on different computer systems in an APPC session.

Cooperative Processing

The ability to access distributed resources in a transparent manner from any system in a network. Systems within a network cooperate with one another in carrying out the distributed access.

CP

See Control Point.

CPI

See Common Programming Interface.

CPIC

See Common Programming Interface for Communications.

CRC

See Cyclic Redundancy Check.

CUA

See Common User Access.

Customer Information Control System (CICS)

A popular IBM timesharing package that runs under several IBM operating systems and is optimized for end-user applications.

Cyclic Redundancy Check

A means of determining whether a message has been received properly. The sender calculates a check sum of each character position in the message, and sends the result at the end of the message. The receiver, performing the same calculation, should get the same result.

Data

Information coded in computer language and having a defined code structure meaningful to both the sender and the receiver.

Data Flow Control (DFC)

The Data Flow Control layer supports the SNA protocols needed to coordinate the flow of data on an SNA session. Protocols are provided to control the direction of data flow and control logical grouping of data for error recovery purposes, to send responses to data sent on the session, and to provide support for other session control functions.

Data Line Monitor

A device used to view the actual data on a communication line, consisting of a CRT display and switches to control the monitoring process. This device is normally used for troubleshooting.

Data Link

See Link.

Data Link Control (DLC)

The Data Link Control layer defines the protocols that SNA uses to transmit data across a single physical data link and to perform error detection and error recovery functions for that data. These DLC protocols are defined by the System/370 channel interface and Token Ring for local area networking. Synchronous Data Link Control (SDLC) is the DLC protocol used across wide area networks.

Database

Logically organized data, which may or may not be recalled through a sorted index.

DDM

See Distributed Data Management.

Descriptor

A data structure used in a calling procedure for passing argument types, addresses, and other information.

DFC

See Data Flow Control.

DIA

See Document Interchange Architecture.

Disk Operating System (DOS)

IBM's standard operating system for the PC family of systems. DOS is a single tasking operating system that runs on PC and PS/2 systems.

DISOSS

See Distributed Office Support System.

Display Station

A CRT terminal or intelligent workstation with a monitor and keyboard that allows information to be entered into the network and has the ability to display information received from the network. This term is typically associated with 3270 display stations (for example, 3179).

Distributed Data Management (DDM)

An architecture describing a form of remote file access in SNA networks. DDM transaction programs on different systems use LU 6.2 to distribute requests for record and file access.

Distributed Office Support System (DISOSS)

An IBM System/370 mainframe-based office application that runs under CICS. DISOSS provides a number of office-oriented services, allowing end users on different systems to exchange mail and documents, to store and retrieve documents from host libraries, and to distribute documents in a network.

Distributed Processing

Processing performed by multiple systems in a network.

Distributed Resource

A resource residing on one system which is being accessed from another system in a network. The resource being accessed is not locally resident on the requesting system.

Distributed Transaction Processing

Distributed processing between cooperating transaction programs.

DLC

See Data Link Control.

Document Interchange Architecture (DIA)

An IBM architecture describing a set of services that allows documents to be exchanged between different systems in an SNA network. DIA services include document library services which allow users to store and retrieve documents to and from host libraries, and document distribution services.

DOS

See Disk Operating System.

DOS/VSE

See Virtual System Extended (VSE).

Dumb Terminal

A terminal, such as a 3270 terminal, that has no application processing capability of its own. It relies solely on host-based application programs for processing and control. Terminal operators use such terminals to access host applications.

EBCDIC

See Extended Binary Coded Decimal Interchange Code.

Editor

A program used to create and modify text files.

End User

A device, such as a display station, a printer, or an application program that is the source or destination of data exchanged in an SNA network. LUs represent end users, serving as their port into and out of the network.

Error Code

A status code that is returned by an operating system service subroutine to indicate that an error has occurred during execution.

Error Message

A character string that is associated with an error code.

Event Flag

Event flags are status posting bits maintained for general programming use. In the APPC/LU 6.2 Programming Interface, an event flag is set to indicate asynchronous completion of a procedure.

Exchange Identification

See Exchange Station ID.

Exchange Station ID (XID)

A code that allows one computer to recognize another computer or device in a dial, Token Ring, or APPN network. Each code must be unique within a network. This is also short for Exchange Station ID, an SDLC supervisory frame in which two nodes first making contact identify themselves. An XID frame contains a four-byte node ID and can also contain optional information such as the NETID and CP name. An XID is optional on leased lines, but is commonly employed over dial-up connections and LANs.

Extended Binary Coded Decimal Interchange Code (EBCDIC)

A code system that allows a binary representation of characters and certain control information. EBCDIC is the standard developed and used by IBM in their mainframe and mini-computers. It differs from ASCII in that it uses all eight bits of a byte to represent a character and therefore may represent more characters (see also American Standard Code for Information Interchange).

File

A set of records or bytes stored on disk or tape as a unit. A disk file has a path name that identifies it as a unique entity in the system's directory hierarchy. Attributes of a disk file, such as its size and when it was created, are maintained in the directory containing the file.

Fixed File

A file with a fixed organization. In a fixed file, the records are the same size. Each record is stored in a disk or tape region holding a number of bytes that is the same for all the records in the file. Compare with relative file, sequential file, and stream file.

FMH

See Function Management Header.

Frame

A more modern term for block, normally applied to protocols that bound their messages by fixed codes, such as SDLC and HDLC.

Full Duplex

A physical connection on which information can be transmitted in both directions between the same two points simultaneously.

Function Management Header (FMH)

Control information that allows an LU to transmit a data stream to a specific destination and control the presentation of the data at that destination. FMHs are the means by which an LU selects the functions it wants the presentation services components of its session partner to perform. For more information about FMHs, see IBM's *Systems Network Architecture - Sessions Between Logical Units,* Order No. GC20-1868.

Functional Layer

One of the major groupings of functions defined by the SNA architecture. Major functions have been isolated into separate layers so that changes can be made to one layer without having to change other layers, thereby making it easier to accommodate new technologies in SNA. SNA consists of seven layers: Transaction Services, Presentation Services, Data Flow Control, Transmission Control, Path Control, Data Link Control, and Physical Control.

Gateway

A system that handles communications between dissimilar link types.

GDS

See Generalized Data Stream.

Generalized Data Stream (GDS)

A standard data stream that is used in LU 6.2 communication. (See also Basic Conversations and Mapped Conversations.)

Half Duplex

A physical connection on which information can be transmitted in both directions, but not simultaneously. While sending, each party has the entire capacity of the channel.

HDLC

See High Level Data Link Control.

Hertz

Cycles per second (CPS).

Hierarchical

A connection characterized by a master-slave relationship. In SNA terminology, a hierarchical connection involves a Peripheral Node (slave) connection to a host subarea node (master). The host node maintains control and is responsible for initiating and terminating the connection.

High Level Data Link Control (HDLC)

A communication protocol for full duplex communication which is used as the basis for the packet switching X.25 protocol. HDLC is a level 2 protocol in terms of the ISO model.

Host Node

A Type 5 SNA node which contains a System Services Control Point (SSCP). A Host Node is typically a System/370 mainframe computer running VTAM. A Host Node serves as a centralized control point of a network. Host Nodes support application programs and provide network management services. A Host Node is one type of subarea node, the other type being a Communications Controller Node.

I/O Status Vector

This mechanism is VAX-specific and provides the VMS transaction program with complete information about error conditions. The status vector provides a top-level completion code, and in many cases, further qualifying codes.

IMS

See Information Management System.

Independent Logical Unit

A logical unit that does not rely on the SSCP to establish its sessions. To be independent, a logical unit must meet these criteria:

- It must be a Logical Unit Type 6.2
- The Physical Unit node it resides on must be Type 2.1
- It must be capable of sending a bind request
- It must support parallel sessions
- Its node must be able to send an XID, which includes a Control Point name vector

Information Management System (IMS)

An IBM System/370 host-based Database/Data Communications (DB/DC) subsystem that runs in the MVS environment. IMS supports user-written batch and interactive applications, providing communications access to remote systems, and access to databases maintained by IMS for these applications.

Initial Program Load (IPL)

The process of transferring the operating system from disk into the main memory of a computer, so that it can prepare itself to run application programs. See also Reboot.

Intelligent Workstation

A system, such as a personal computer, that has its own processing capability. Local applications can execute on an intelligent workstation so that connection to a host system is not required.

Interface

The connection between a computer or terminal and a datapath. There are various interface standards, including RS232.

Intermediate Node

A node in a network that provides network routing services but is not the source or destination of the data being routed. An intermediate node receives data from one node and, based on addressing information, sends the information to another node. Host Nodes and Communications Controller Nodes can function as intermediate nodes in SNA networks.

International Standards Organization (ISO)

An organization responsible for issuing recommendations on various issues in computers and communication. The ISO developed the famous seven layer Basic Reference Model for Open Systems Interconnection, called the ISO Model.

Intersystem Communication (ISC)

The method CICS uses to enable communication between separate systems by means of programmable interfaces and SNA. (See also CICS.)

IPL

See Initial Program Load.

ISC

See Intersystem Communication.

ISO

See International Standards Organization.

JES

See Job Entry Subsystem.

Job Entry Subsystem (JES)

A series of IBM products (JES, JES2, JES3) that allows remote users to submit jobs for execution on System/370 mainframes. The JES subsystems run in the MVS operating system environments.

LAN

See Local Area Network.

LEN

See Low Entry Networking.

Level N

One of the seven levels or layers of the ISO model. Level 1 is the physical interface, and level 7 is the application or user interface.

Link

A physical connection between two nodes. Different types of links are supported in SNA networks, including System/370 local channel links, Token Ring LANs, and SDLC links. The Data Link Control (DLC) layer defines various protocols to manage data exchange across the different links.

Link Protocol

The set of rules which govern communications over a single link within a network. The link or data link protocol is Level 2 of the ISO Model.

Local Area Network (LAN)

A communication technology which passes information over relatively short distances (less than one mile, typically) and at very high speeds. (This term is usually pronounced as a word, "lan," rather than by pronouncing the letters separately.)

Local Channel

The direct link between one IBM System/370 processor and either another System/370 processor, a Communications Controller, or a Peripheral Node. The local channel interface is one of the data links supported in SNA networks.

Logical Link

A temporary conversation path established between two transaction programs in a network.

Logical Unit (LU)

The addressable entity in an SNA network, with which a user can send and receive messages. An LU could also be described as a user's portal to the network. An LU can be implemented in either hardware, software, or firmware. Whatever fulfills the SNA responsibilities of the LU is said to implement the LU.

Logical UnitToLogical Unit (LULU) Session

A logical connection between the LUs representing users of an SNA network. No communications can occur between LUs until they establish an LU-LU session.

Low Entry Networking (LEN)

A relatively new technical direction within SNA that provides for the routing of traffic between any two nodes, automatic definition of routes, and directory services. The enhancement to SNA that makes Low Entry Networking possible is called the Type 2.1 Network Node.

LU

See Logical Unit.

LU Type 0

An implementation-defined LU type that is generally used to support program-to-program communications. It is most often used in IBM's industry-specific systems (for example, retail POS, banking), and is being superseded by LU Type 6.2 in most new applications.

LU Type 1

The LU type that is designed to support communications between a remote terminal and a host-based application. LU Type 1 is used for sending SNA Character String (SCS) data streams to 3270 printers. It is also used to support 3770 Remote Job Entry (RJE) terminals.

LU Type 2

The LU type that is designed to support communications with display stations using the 3270 data stream format.

LU Type 3

The LU type that is designed to support communications with printers using the 3270 data stream format.

LU Type 4

An early SNA LU type used for peer-to-peer communications. LU 4 has now been superseded by LU 6.2 for peer-to-peer communications.

LU Type 6.1

The LU type that is designed to support program-to-program communications between IBM host-based applications such as CICS and IMS.

LU Type 6.2

The LU type that is designed to support generalized program-to-program communications. LU Type 6.2 defines an application program interface which applications use to communicate with each other. Another important feature of LU Type 6.2 is its ability to support peer-to-peer sessions directly between users on SNA workstations and mid-range processors.

Mainframe

A System/370 host computer. A mainframe is any model of IBM's System/370, 303x, 308x, 309x, and 4300 and 9370 series of processors. Due to VTAM software which runs in the mainframe, the mainframe functions as a Type 5 Host Node in an SNA network. In addition to serving as a network control point, mainframes also support network management facilities and application programs.

Mapped Conversations

Conversations that take place between user-written transaction programs. Mapped conversations transform all data being sent to another transaction into the generalized data stream (GDS) and then restore the data to its original form before the destination program receives it. In the VAX environment, the VMS transaction program does not recognize GDS headers.

Midrange System

One of IBM's non-mainframe, non-PC systems. Typically, midrange systems fall between mainframes and PCs in their processing power and capacity. IBM midrange systems include the AS/400, System/36, System/38, Series/1, and 8100.

Mode

A set of characteristics used to define a conversation.

Mode Entry

An entry in a VTAM table that describes the mode being used. A mode entry defines characteristics of a session that both sides must agree upon, such as the LU type and the data frame size.

Mode Table

A VTAM module that contains one or more mode entries.

Modem

Modulator/Demodulator. A device which converts digital information to a series of tones suitable for transmission across an analog path, such as a conventional telephone circuit.

MultiDomain SNA Network

An SNA network whose resources (LUs, PUs, and data links) are controlled by more than one SNA host. The Systems Services Control Point (SSCP) within each of the hosts controls some portion of the network's resources.

Multidrop

A wide area communication line which connects more than two points. On multidrop lines, one station is normally designated the master, to prevent interference if two stations attempt to send at once.

Multiple Virtual Storage (MVS)

The general name given to the flagship operating system used on large IBM mainframes. The current version of MVS is z/OS.

Multiplexer

A communication device which combines several user channels into a single trunk line for more economical transmission.

MVS

See Multiple Virtual Storage.

NAU

See Network Addressable Unit.

NCP

See Network Control Program.

NetView

The IBM software package that provides VTAM network management capabilities. NetView provides facilities to allow network operators to monitor and control the network, detect and isolate problems in the network, determine the status of network components, and activate and deactivate network resources.

NetView Distribution Manager

An IBM System/370-based program product that is used for distributing software and microcode to remote systems in an SNA network.

NetView/PC

An IBM program product that runs on PC or PS/2 systems. NetView/PC is used to provide network management support for non-SNA devices. Applications that interface with NetView/PC are required to support the non-SNA device. These applications can then use the services of NetView/PC to send alert messages to NetView on a System/370 host.

Network

A communications facility that connects two or more points.

Network Addressable Unit (NAU)

The origin or the destination of information transmitted through the path control network. In Systems Network Architecture (SNA), it is a Logical Unit, Physical Unit, or Systems Services Control Point.

Network Control Program (NCP)

The operating system for IBM front-end processors. A front end processor cannot be "booted" with NCP until it is first customized during NCP generation.

Network Management

The set of functions and processes used to control a network. Network management functions include activating and deactivating network resources, monitoring the status of resources, detecting and isolating problems in the network, distributing software to systems in the network, configuring the network, and other related activities. NetView is IBM's primary System/370 host-based network management software.

Node

An endpoint of a communications link or a junction common to two or more links in a Network. Nodes can be processors, controllers, or workstations. It became popular a few years ago to stop using the term Physical Unit (PU) and use the term node instead. In SNA, a node embodies the set of responsibilities that govern physical attachment to the network. Whatever fulfills these responsibilities is said to implement the Physical Unit, which can be hardware, software, or firmware.

Node Type

An SNA node that has a particular role in an SNA network. SNA defines four different types of nodes: Host (Type 5) Nodes, Communications Controller (Type 4) Nodes, and Type 2.0 and Type 2.1 Peripheral Nodes.

NonSwitched Line

A communications connection that is not initiated by dialing into a public data network. This is often known as a leased line, referring to the procedure of leasing a dedicated circuit from a phone company. See also Switched Line.

Nucleus

The portion of z/OS code in which very critical, low-level routines reside. Code residing in the nucleus usually cannot be altered without shutting down a computer.

Operand

A parameter used to define the meaning of a macro instruction. A Network Control Program, for example, is customized using macros, and it is common that, as new features are added, new operands must be specified to go with them.

Organization (File)

The way records in a file are stored on a disk. The four types of organization are sequential, relative, fixed, and stream.

OS/2

A multitasking operating system for IBM's PS/2 family of personal computers. OS/2 will also run on some models of the PC family, such as PC AT and PC XT 286 systems.

OS/2 EE

See OS/2 Extended Edition.

OS/2 Extended Edition

IBM's version of OS/2 that includes an integrated relational database manager and an integrated communications manager. OS/2 EE is the SAA environment for intelligent workstations.

Outbound Primary Logical Unit

A peripheral node that can start a session by sending a bind request. Outbound PLUs allow two non-host computers to hold sessions over an SNA backbone network, a capability that was only recently made possible.

Pacing

A method of limiting the number of frames sent between two systems to prevent the data buffer of the receiving machine from becoming overrun.

Packet Assembly/Disassembly (PAD)

A hardware device which converts information between the X.25 packet protocol format and a non-packet protocol so that non-packet devices may use a packet network.

Packet Switching

A form of communication where multiple users place their data on a single trunk line to minimize the cost per user. Each conversation is identified by a logical channel number. The X.25 protocol is a CCITT standard for packet switching.

PAD

See Packet Assembly/Disassembly.

Parallel

A data transmission method where the bits in a character are sent on eight channels simultaneously rather than on a single channel. Printers and other local devices are often interfaced via parallel transmission, but this method is rare in data communication.

Password

A sequence of characters that a user can be required to supply for security reasons when logging onto a system. The characters in a password do not appear on the screen when they are typed in.

Path Control

Layer 3 of SNA that deals with the routing of data through a network. Path Control provides end-to-end network routing that may span multiple data links. Path Control is responsible for setting up an end-to-end path through the network connecting a pair of end users.

PC

IBM's original family of personal computer systems. PCs are based on Intel microprocessors including the 8088, 8086, and 80286 chips. DOS is the primary operating system running on most PCs. The original PC line was enhanced to include PC XT and PC AT models. The PC family has been superseded by the PS/2 family.

Peer-to-Peer Communications

Communications between two adjacent Type 2.1 Nodes. This type of peer connectivity is called Low Entry Networking (LEN). Such communications do not require a System/370 mainframe to be involved. This is contrasted with hierarchical communications in which a Type 2.0 or Type 2.1 Peripheral Node must be connected to a host system. Only LU 6.2 sessions are supported across peer-to-peer connections between Type 2.1 Nodes.

Peripheral Node (PN)

A Type 2.1 node that contains Logical Units, one of which is a session partner. Peripheral Nodes support the attachment of end users to the network. End users may be devices, such as displays and printers, or they may be application programs. While Type 2.0 Nodes are restricted to connecting to Host Nodes or Communication Controller Nodes, Type 2.1 Nodes can also directly connect to other Type 2.1 Nodes without host involvement.

Physical Control Layer

Layer 1 of SNA that deals with the physical connections of systems into the network. IBM supports a number of industry standard physical interfaces at this layer, including the RS-232 and 802.2 Token Ring interfaces. This layer addresses the mechanical and electrical interfaces used to get information into and out of a system.

Physical Unit

A synonym for node. See also Node.

Physical Unit Name (PU name)

The mnemonic name given to a node. This name is defined to VTAM and/or NCP, and sometimes to the node itself.

Physical Unit Type 2

Now retroactively called Physical Unit 2.0, this is the specification for a Peripheral Node; examples of PU 2.0 implementations include 3174 cluster controllers and most minicomputers. Two Type 2.0 PUs cannot attach to each other, but instead must be connected to a larger node (PU Type 4 or 5). Node Type 2 is being phased out.

Physical Unit Type 2.1

IBM's strategic node type. A PU 2.1 node can be directly connected to any other PU 2.1 node. All future IBM products will be Type 2.1 nodes.

Physical Unit Type 4

The node type of a front-end processor; this is being phased out. A node Type 4 is subordinate to a node Type 5.

Physical Unit Type 5

The node type of a host device that contains a System Services Control Point. To implement a PU Type 5 is to emulate a mainframe.

PIP

See Program Initialization Parameter.

PLU

See Primary Logical Unit.

PN

See Peripheral Node.

Poll

A periodic request by a master communication station to a slave station, requesting that the slave provide data or status information. Polling is the act of requesting status information from multiple stations.

Presentation Services (PS)

Layer 6 of SNA that deals with the presentation of data to and from end users. For older style terminal emulation (e.g., 3270), PS deals with the formatting and unformatting of data to and from a format that is usable by end users such as displays and printers. For LU 6.2, PS also provides an application program interface (API) used by application programs to access the services provided by the LU.

Presentation Services Profile

A subset of SNA Presentation Services (PS) protocols supported by a particular type of LU.

Primary Logical Unit (PLU)

An LU that initiates an SNA session with a partner LU. For each SNA LU-LU session, one LU assumes the role of the primary LU while the partner LU assumes the role of the secondary LU. The primary LU can initiate and terminate the session by sending the SNA BIND and UNBIND commands.

Procedure

A routine entered by means of a call in the transaction program.

Process

The sequence of states of the hardware and software during the execution of a user's program.

Profile

A subset of SNA commands and protocols at a particular layer of the SNA architecture.

Program Initialization Parameter (PIP)

The means of passing program initialization parameters to the destination transaction program.

Program Temporary Fix (PTF)

A patch sent by the vendor to fix a bug, or bugs, in a software package.

ProgramtoProgram Communications

Communications between a pair of programs. LU 6.2 provides program-to-program communications protocols. Two Transaction Programs (TPs) can communicate with one another using LU 6.2 protocols. IBM refers to this as Advanced Program-To-Program Communications (APPC).

Protocol

A set of agreements by which two or more stations agree on information structures, codes, and so on, required for successful and error-free communication. SDLC, X.25, and so on are protocols. Data link control protocols, such as SDLC, define the rules of interaction between two nodes connected by a single data link.

Protocol Boundary

The interface between a Transaction Program (TP) and a Type 6.2 LU. The protocol boundary consists of a set of verbs which describe the functions provided by the LU that are available to the TP.

PS

See Presentation Services.

PTF

See Program Temporary Fix.

Queue

A mechanism to record jobs that are waiting to be processed. Jobs can be given priorities within a queue.

Reboot

The startup of a computer, when the operating system is read in from a disk and necessary preparations are made for applications to be run.

Reference

An argument in an APPC/LU 6.2 Programming Interface procedure that is passed by an address and is usually expressed as a reference name or label associated with a particular area or field.

Relative File

A file with a relative organization. In a relative file, the records can have varying sizes. Each record is stored in a disk or tape region holding a number of bytes that is the same for all the records in the file. Compare with fixed file, sequential file, and stream file.

Remote IBM Host Transaction Program

The transaction program residing on the IBM system with which another transaction program communicates.

Remote Job Entry (RJE)

Originally, the name given to old-fashioned card reader/line printer devices that were attached to a mainframe via telephone lines. Today, the term RJE is used to describe small computers that, in order to communicate with a host, emulate those old printer/readers. RJE communications use an old SNA protocol called Logical Unit Type 1.

Restore

A command used to copy onto disk data that was written to tape by the save command.

RJE

See Remote Job Entry.

RS232

A standard for computer/terminal interfacing which defines a 25 pin connector. RS232 is the most popular interface standard.

SAA

See Systems Application Architecture.

SCS

See SNA Character String.

SDLC

See Synchronous Data Link Control.

Secondary Logical Unit (SLU)

The LU that accepts a session establishment request from a primary LU. For each SNA session between a pair of LUs, one LU functions as the primary LU while the other LU functions as the secondary LU. The primary LU is responsible for initiating (with the SNA BIND command) and terminating (with the SNA UNBIND command) the LU-LU session, while the secondary LU accepts or denies these requests.

Sequential File

A file with a sequential organization. In a sequential file, the records can have varying sizes and each record is stored in a disk or tape region holding approximately the same number of bytes as in the record. Thus, the record storage regions in a sequential file vary from record to record. Compare with fixed file, relative file, and stream file.

Serial

A data transmission method where the bits in a character are sent one after the other over a single channel. Most modern data communication is serial.

Session

A logical connection that permits communication between two logical units. Before any communication is possible between a pair of NAUs, a session must be established that logically connects the pair. SNA supports SSCP-SSCP, SSCP-PU, SSCP-LU, and LU-LU sessions. SNA session level protocols are used to manage the exchange of information across the session between the pair of NAUs.

Single Domain Network

An SNA network with only one System/370 host, and therefore only a single System Services Control Point (SSCP) (which is implemented in VTAM in the System/370 host). All resources in the network are controlled by the single host.

SLU

See Secondary Logical Unit.

SMP/E

See System Modification Program/Extended.

SNA

See Systems Network Architecture.

SNA Character String (SCS)

A data stream consisting of intermixed end-user data, single byte control codes, and/or multi-byte control sequences. The control codes and sequences are used to direct the presentation of the end-user data on display stations or printers. SCS is the type of data stream used with LU Type 1 processing.

SNA Network

Multiple interconnected systems using SNA protocols between systems to provide support for end-to-end information exchange. Typically, an SNA network consists of one or more System/370 mainframes that act as central control points, multiple communications controllers that act as intermediate routing nodes, and many terminals, PCs, and multi-user systems that support end users of the network.

SNA/Distribution Services (SNA/DS)

An architecture that describes a generalized, asynchronous delivery facility based on store-and-forward techniques. SNA/DS is used to distribute different types of objects (for example, mail, documents, programs) through an SNA network. SNA/DS is used in conjunction with Document Interchange Architecture (DIA) to provide electronic mail services. SNA/DS uses LU 6.2 for its program-to-program communications.

SNA/DS

See SNA/Distribution Services.

SNA/Management Services

IBM's set of network management services that are used for managing systems in an SNA network. NetView is IBM's major host-based product that supports SNA/Management Services.

SSCP

See System Services Control Point.

SSP

See Systems Support Program.

Start/Stop Bit

The bits that mark the beginning and end of each asynchronously transmitted character.

State

The condition of a conversation at a particular point in time. For example, when a conversation is in receive state, the transaction program cannot send data; it can only receive data. The state of a conversation determines what procedures a transaction program can call.

Stream File

A file with a sequential organization. In a stream file, the records can have varying sizes and each record is stored in a disk or tape region holding approximately the same number of bytes as in the record. Thus, the record storage regions in a stream file vary from record to record. In these ways, stream files are similar to sequential files.
However, stream files differ from sequential files in the following ways:
While a sequential file must be accessed on a record basis, a stream file can be accessed on either a record or byte basis.
Sequential files are stored on disk with the record size at the beginning and end of each record. Stream files do not have any record size information stored with them; each new line character in the file is interpreted as the end of a record.
It is possible to have a stream file with only one record. When stream files are used to store text, each record contains one line of text. Compare with fixed file, relative file, and sequential file.

Subarea

A division of an SNA network that is under the control of, and addressed by, a front-end processor or a host access method such as VTAM.

Switched Line

A connection between two or more points that requires the user to dial into a public data network.

Synchronous

A form of transmission where the sender and receiver exchange clocking or timing information and send a block or frame with no space or markings between characters. Because no start/stop bits are required, synchronous transmission is about 25% more efficient than asynchronous transmission.

Synchronous Communications

Transmission of data at a fixed rate, with the transmitter and receiver synchronized by means of synchronization characters located at the beginning of each message or block of data.

Synchronous Data Link Control (SDLC)

SDLC is the synchronous protocol used by SNA. It is level 2 or the link level protocol of IBM's SNA. SDLC is essentially a subset of HDLC. It provides buffering and polling capabilities. CA XCOM Data Transport runs over SDLC.

System Modification Program/Extended (SMP/E)

An IBM-supplied utility program that is used to facilitate the installation of software packages and patches. SMP/E helps ensure that programs are installed in the proper sequence and that all prerequisites and corequisites are met.

System Services Control Point (SSCP)

The entity that controls and manages the resources of a data communications network that are owned by a host.

System/36

An IBM multi-user, midrange applications processing system. It supports a wide range of SNA communications, including 3270 emulation and peer-to-peer communications using Logical Unit Type 6.2. The System/36 is being replaced by IBM's AS/400 system.

System/370

IBM's family of mainframe processors that are based on the System/370 architecture. Included in this family are 308X, 3090, 4300, and 9370 processors.

System/38

An IBM minicomputer and the forerunner of the AS/400; noted for the relational database integrated into its operating system, the System/38 is hindered by its 1970s hardware design.

Systems Application Architecture (SAA)

IBM's umbrella architecture that consists of components used to bring greater consistency to IBM mainframe, midrange, and workstation products. SAA covers user interface, programming languages and services, and communications issues.

Systems Network Architecture (SNA)

The logical description of all components within IBM networking strategy and how they interact. It describes sets of protocols used for communications between logical components. SNA also defines a set of layers that describe different levels of functionality needed to support communications between end users. CA XCOM Data Transport is an SNA-based software package.

Systems Support Program (SSP)

The utility programs that load, dump, and debug the Network Control Program. (SSP here is short for the mainframe ACF/SSP, and is not to be confused with the SSP operating system of the System/36.)

TC

See Transmission Control.

Text File

A sequential or stream file containing only ASCII characters.

Third-Party Processing

(A CA term) Use of a computer system to arrange for a second computer system to send a file, job, or report to a third computer system, at a specified future time if not immediately.

Throughput

In data communications, the amount of data transmitted through a communications link in a fixed period of time.

Time-Sharing Option (TSO)

A very powerful (but high-overhead) IBM interactive application that is used by programmers and operators both to develop software and to customize and control a z/OS system.

Token Ring

IBM's strategic local area network (LAN) based on a token passing scheme that is compatible with the IEEE 802.2 and 802.5 Token Ring standards. A Token Ring LAN interface is one of the data link types supported in SNA networks.

TPN

See Transaction Program Name.

Transaction Program

An IBM-supplied or user-written application program that directly interfaces to a Type 6.2 LU via the LU 6.2 application program interface (API) provided by the LU. A logical connection, called a conversation, must be set up between the Transaction Programs (TPs) before any communication can take place. The conversation and exchange of information between the TPs is carried out over an underlying SNA LU 6.2 session managed by the pair of LUs supporting the TPs. Examples of IBM-supplied TPs are DIA, SNA/DS, and DDM.

Transaction Program Name (TPN)

The identifier of the remote IBM host transaction program with which another transaction program wants to engage in an LU 6.2 conversation.

Transaction Services (TS)

Layer 7 of SNA that deals with application level processing. Programs running at the Transaction Services layer are called Transaction Programs (TPs). These may be either IBM-supplied TPs such as DIA, SNA/DS, and DDM or they may be user-written TPs. TPs interact directly with a Type 6.2 LU via an LU 6.2 application program interface (API).

Transmission Control (TC)

Layer 4 of SNA that deals with initiating and terminating sessions, controlling the rate of flow of information across the session, checking sequence numbers to ensure that information is not lost or duplicated, and encrypting and decrypting data as required.

Transmission Subsystem Profile

A subset of SNA Transmission Control (TC) commands and protocols supported by a particular type of LU.

TS

See Transaction Services.

TSO

See Time-Sharing Option.

Type 2.0 Node

An SNA Peripheral Node that is capable of only hierarchical connections with a host system. Examples of Type 2.0 Node products are 3270 cluster controllers and other systems emulating 3270 operation.

Type 2.1 Node

An SNA Peripheral Node that supports both hierarchical connections to Host Nodes as well as peer-to-peer connections to other Type 2.1 Nodes with no host involvement. Examples of Type 2.1 Nodes are AS/400s, System/36s, System/38s, and PCs. APPC software in these systems supports the Type 2.1 Node facilities.

Type 4 Node

An SNA Communications Controller Node that serves as an intermediate routing node in an SNA network. Type 4 Node functions are implemented by Network Control Program (NCP) software running in a 37xx Communications Controller. The primary functions of a Type 4 node are network routing and managing physical data links connected to the node.

Type 5 Node

An SNA Host Node characterized by a System Services Control Point (SSCP). A Type 5 Host Node is typically a System/370 mainframe running VTAM. It supports application program end users and provides network management services.

V.35

The most popular interface standard for data connections over 19.2 KBPS, defined by the CCITT.

Verbs

The LU functions available to Transaction Programs (TPs) that are formally architected as part of the LU 6.2 Protocol Boundary.

Virtual Machine (VM)

One of IBM's primary System/370 operating systems. VM presents a virtual machine interface to users so that, to each user, it appears that the entire machine is dedicated to their use.

Virtual System Extended (VSE)

A widely used IBM mainframe, operating system, not covered by IBM's Systems Application Architecture, as are MVS and VM. Formally called DOS/VSE, but VSE is a widely accepted shorter name.

Virtual Telecommunications Access Method (VTAM)

The software package that mainframe applications use to send and receive messages over an SNA network; implements Layers 2, 3, and 4 of SNA.

VM

See Virtual Machine.

VMS Transaction Program

The VMS application that the user writes using CA XCOM Data Transport.

VMS/SNA

VMS/SNA is a single-system connection to an IBM SNA network. It consists of the communications software necessary to allow a user of a MicroVAX I, MicroVAX II, VAXstation II, or VAXBI family of systems to connect directly to and participate in an SNA network.

VPS

See VTAM Printer Support.

VTAM

See Virtual Telecommunications Access Method.

VTAM Printer Support (VPS)

A software package (including a VTAM application) from Levi, Ray, and Shoup that sends spooled output to 3270 attached printers.

XCOM

CA XCOM Data Transport
A software package (including a VTAM application) from CA Technologies that provides for bulk data transfer between mainframes, minis, and micros using LU 6.2.

XID

See Exchange Station ID.

Zap

Short for SuperZap, this is an omnipotent utility program that allows those authorized to use it to modify external storage without limit. SuperZap can be dangerous, and its use is typically reserved to systems programmers.

Index

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X