

PMDF Obsolete Features

Order Number: PMDF-OBS-65

February 2010

This document describes obsoleted features in version 6.5 of PMDF.

Revision/Update Information: This manual supersedes the V6.4 *PMDF Obsolete Features*

Software Version: PMDF V6.5

Operating System and Version: Solaris SPARC or x86 V2.6, V8 or later; (SunOS V5.6, V5.8 or later);
Tru64 UNIX V4.0D or later;
Red Hat Enterprise Linux 4 update 8 or later on x86; (or other compatible Linux distribution)
OpenVMS Alpha or VAX V6.1 or later;
OpenVMS I64 V8.2 or later;
Windows 2000; Windows 2003

Copyright ©2010 Process Software, LLC.
Unpublished — all rights reserved under
the copyright laws of the United States

No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means electronic, mechanical, magnetic, optical, chemical, or otherwise without the prior written permission of:

Process Software, LLC
959 Concord Street
Framingham, MA 01701-4682 USA
Voice: +1 508 879 6994; FAX: +1 508 879 0042
info@process.com

Process Software, LLC ("Process") makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Furthermore, Process Software reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Process Software to notify any person of such revision or changes.

Use of PMDF, PMDF-DIRSYNC, PMDF-FAX, PMDF-LAN, PMDF-MR, PMDF-MSGSTORE, PMDF-MTA, PMDF-TLS, PMDF-X400, PMDF-X500, PMDF-XGP, and/or PMDF-XGS software and associated documentation is authorized only by a Software License Agreement. Such license agreements specify the number of systems on which the software is authorized for use, and, among other things, specifically prohibit use or duplication of software or documentation, in whole or in part, except as authorized by the Software License Agreement.

Restricted Rights Legend

Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or as set forth in the Commercial Computer Software — Restricted Rights clause at FAR 52.227-19.

The PMDF mark and all PMDF-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries and are used under license.

ALL-IN-1, Alpha AXP, AXP, Bookreader, DEC, DECnet, HP, I64, IA64, Integrity, MAILbus, MailWorks, Message Router, MicroVAX, OpenVMS, Pathworks, PSI, RMS, TeamLinks, TOPS-20, Tru64, TruCluster, ULTRIX, VAX, VAX Notes, VMScluster, VMS, and WPS-PLUS are registered trademarks of Hewlett-Packard Company.

AS/400, CICS, IBM, Office Vision, OS/2, PROFS, and VTAM are registered trademarks of International Business Machines Corporation. CMS, DISOSS, OfficeVision/VM, OfficeVision/400, OV/VM, and TSO are trademarks of International Business Machines Corporation.

dexNET is a registered trademark of Fujitsu Imaging Systems of America, Inc.

FaxBox is a registered trademark of DCE Communications Group Limited.

InterConnections is a trademark of InterConnections, Inc.

LANmanager and Microsoft are registered trademarks of Microsoft Corporation.

MHS, Netware, and Novell are registered trademarks of Novell, Inc.

PGP and Pretty Good Privacy are registered trademarks of Pretty Good Privacy, Inc.

Attachmate is a registered trademark and PathWay is a trademark of Attachmate Corporation.

PostScript is a registered trademark of Adobe Systems Incorporated.

SPARC is a trademark of SPARC International, Inc.

UNIX is a registered trademark of UNIX System Laboratories, Inc.

Gold-Mail is a trademark of Data Processing Design, Inc.

Copyright (c) 1990, 1993, 1994, 1995 The Regents of the University of California. All rights reserved.

AlphaMate is a registered trademark of Motorola, Inc.

cc:Mail is a trademark of cc:Mail, Inc., a wholly-owned subsidiary of Lotus Development Corporation. Lotus Notes is a registered trademark of Lotus Development Corporation.

RC2 and RC4 are registered trademarks of RSA Data Security, Inc.

Ethernet is a registered trademark of Xerox Corporation.

GIF and "Graphics Interchange Format" are trademarks of CompuServe, Incorporated.

InterDrive is a registered trademark of FTP Software, Inc.

Memo is a trade mark of Verimation ApS.

LaserJet and PCL are registered trademarks of Hewlett-Packard Company.

Jnet is a registered trademark of Wingra, Inc.

Pine and Pico are trademarks of the University of Washington, used by permission.

Solaris, Sun, and SunOS are trademarks of Sun Microsystems, Inc.

TCPware and MultiNet are registered trademarks of Process Software.

TIFF is a trademark of Aldus Corporation.

Copyright (c) 1990-2000 Sleepycat Software. All rights reserved.

Copyright (c) 1995, 1996 The President and Fellows of Harvard University. All rights reserved.

Contents

PREFACE

vii

CHAPTER 1	COMMAND LINE CONFIGURATION UTILITIES	1-1
<hr/>		
CHAPTER 2	OBSOLETE CHANNEL KEYWORDS	2-1
2.0.1	Gateway daemons (daemon) _____	2-1
2.0.2	Multiple Gateways on a Single Channel (multigate, nomultigate) _____	2-1
2.1	BATCH SMTP CONTINUATION LINES (CONTCHAR, CONTPOSITION)	2-2
2.2	USING LOCAL USERNAMES AS USER TAGS (LOCALUSER, NOLOCALUSER)	2-2
2.3	GENERATION OF TICKet BSMTP COMMANDS (TICK, NOTICK)	2-3
2.3.1	Originating User Tag (user) _____	2-3
2.4	GENERATION OF VERBoSe BSMTP COMMANDS (VERB_ON, VERB_OFF, VERB_NONE, VERB_NEVER)	2-4
<hr/>		
CHAPTER 3	BITNET CHANNELS (OPENVMS)	3-1
3.1	THE NJE ENVIRONMENT	3-1
3.2	THE BITNET ENVIRONMENT	3-2
3.3	SETTING UP A LOCAL Jnet CHANNEL	3-2
3.3.1	Adding the Channel to the Configuration File _____	3-3
3.3.2	Channel Naming Conventions _____	3-3
3.4	SETTING UP A LOCAL ANJE CHANNEL	3-3
3.4.1	Relinking the USERNAME Image _____	3-4
3.4.2	Adding the Channel to the Configuration File _____	3-4
3.4.3	Channel Naming Conventions _____	3-4
3.5	USING AN OPTION FILE TO CUSTOMIZE NJE CHANNELS	3-4
3.6	DEFINING THE SYSTEMS ACCESSIBLE VIA NJE	3-6
3.7	SETTING UP ACCESS TO BITNET MAILER GATEWAYS	3-6
3.8	BITNET GATEWAY TABLES AND BITNET_DOMAINS_DRIVER.COM	3-7
3.8.1	Operating as a Trusted Mailer on BITNET _____	3-12
3.8.2	Using the verb_on and tick Keywords _____	3-12
3.8.3	Using the Multigate Keyword _____	3-12
3.9	MAINTENANCE OF PMDF'S BITNET ROUTING TABLES	3-13
3.10	SUPPRESSING PROGRESS MESSAGES	3-14
3.11	INSTALLING PMDF AS THE LOCAL MAIL DELIVERY (LMD) AGENT	3-14
3.11.1	Jnet Local Mail Delivery _____	3-14
3.11.2	ANJE Local Mail Delivery _____	3-16
3.11.3	Envelope Creation in BN_SLAVE with Jnet _____	3-16
3.11.4	Handling Usernames Longer Than Eight Characters _____	3-17

Contents

3.12	ROUTING TO SYSTEMS NOT RUNNING NJE	3-18
3.13	LOCAL NJE MAILERS AND MAILER GATEWAYS	3-20
3.13.1	Local Mailer Operation	3-20
3.13.2	Setting Up a Local Mailer	3-20
3.13.3	Registering a Mailer or Mailer Gateway on BITNET	3-21
3.13.4	Registering Your NJE Gateway to Serve an Entire Domain	3-23
<hr/>		
CHAPTER 4	PMDF-XGS	4-1
4.1	REQUIRED HARDWARE AND SOFTWARE VERSIONS	4-1
4.2	BACKGROUND SNADS CONCEPTS	4-2
4.2.1	SNADS Architecture	4-2
4.3	PMDF-XGS ARCHITECTURE	4-3
4.3.1	SNADS Addresses in PMDF-XGS	4-3
4.3.2	The Topology of the PMDF-XGS Connection to the SNADS World	4-4
4.3.3	The Function of the Transport Bridge	4-4
4.3.4	Message Attachments	4-5
4.3.5	Notification/status Messages	4-5
4.4	ADDRESSING SOLUTIONS	4-6
4.5	EXAMPLE SITE USING PMDF-XGS	4-6
4.6	CONFIGURING PMDF-XGS	4-8
4.6.1	Configuring a DISOSS Node	4-9
4.6.1.1	Adding Directory Information on a DISOSS Node	4-10
4.6.1.2	Adding Routing Information on a DISOSS Node	4-10
4.6.1.3	Defining the SNA Link in VTAM and CICS on an Adjacent DISOSS Node	4-12
4.6.1.3.1	The VTAM Definition for the SNA Link on DISOSS	4-12
4.6.1.3.2	The CICS Definition for the SNA Link on DISOSS	4-13
4.6.2	Configuring an AS/400 Node	4-14
4.6.2.1	Adding Directory Information to an AS/400 Node	4-15
4.6.2.2	Adding Routing Information on an AS/400 Node	4-16
4.6.2.3	Defining the SNA Link on an Adjacent AS/400 System	4-18
4.6.3	Configuring the PMDF-XGS Transport Bridge	4-19
4.6.3.1	Configuring the SNA Connection on the PMDF-XGS Transport Bridge	4-19
4.6.3.1.1	Parameters for the Local Node Definition	4-20
4.6.3.1.2	Parameters for the SNA Connection Definition	4-21
4.6.3.1.3	Parameters for the Partner LU Definition	4-21
4.6.3.1.4	Parameters for the CPI-C Side Information (symbolic destination)	4-22
4.6.3.1.5	Parameters for the Transaction Program Defaults Definition on OS/2	4-22
4.6.3.1.6	OS/2 Transport Bridge Node Definition File Format	4-23
4.6.3.2	Configuring the Programs on the NT PMDF-XGS Transport Bridge	4-25
4.6.3.2.1	Parameters for the NT PMDF-XGS Transport Bridge Programs	4-26
4.6.3.3	Configuring the Programs on the OS/2 PMDF-XGS Transport Bridge	4-26

4.6.3.3.1	Parameters for the OS/2 PMDF-XGS Transport Bridge Programs • 4–27	
4.6.3.4	Configuring TCP/IP on an OS/2 PMDF-XGS Transport Bridge • 4–28	
4.6.4	Configuring PMDF-XGS on the PMDF System _____	4–29
4.6.4.1	Adding the Channel to the Configuration File • 4–29	
4.6.4.2	Channel Option File • 4–30	
4.6.4.2.1	Location of the Option File • 4–30	
4.6.4.2.2	Format of the Option File • 4–31	
4.6.4.2.3	Contents of the Option File • 4–31	
4.6.4.2.4	Example Option File • 4–32	
4.6.4.3	Defining the Service • 4–32	
4.6.4.4	An Alias for Sending to the Addressing Channel • 4–33	
4.6.4.5	Adding MX Records for the SNADS Pseudodomains • 4–33	
4.7	DETAILED EXAMPLE CONFIGURATION	4–34
4.8	MULTIPLE CONNECTIONS USING MULTIPLE SNADS_ CHANNELS	4–43
4.8.1	Configuring the Additional Adjacent SNADS Nodes _____	4–44
4.8.2	Adding Additional Definitions and Servers on the PMDF-XGS Transport Bridge _____	4–44
4.8.2.1	Adding Additional Definitions on the PMDF-XGS Transport Bridge • 4–44	
4.8.2.2	Adding Additional Servers on the PMDF-XGS Transport Bridge • 4–44	
4.8.3	Adding Additional Channels to the PMDF Configuration _	4–45
4.8.3.1	Adding the Channel Definitions and Rewrite Rules • 4–45	
4.8.3.2	Additional Channel Option Files • 4–46	
4.8.4	Example Configuration with Multiple snads_ Channels _	4–46
<hr/>		
CHAPTER 5	THE MULTIWARE CLIENT API APPROACH IN PMDF-LAN (OPENVMS)	5–1
5.1	DIRECT OpenVMS CLIENT ACCESS TO A NETWARE FILE SERVER	5–1
5.2	CONFIGURING USE OF THE MULTIWARE CLIENT API APPROACH IN PMDF-LAN CHANNELS	5–2
<hr/>		
CHAPTER 6	MULTINET SNMP SUBAGENT FOR PMDF (OPENVMS)	6–1
<hr/>		
CHAPTER 7	MULTINET MM USER AGENT (OPENVMS)	7–1
7.1	SENDING MESSAGES WITH MultiNet MM	7–1
7.2	RECEIVING MESSAGES WITH MultiNet MM	7–1
7.3	THE PMDF OPTION FILE AND MULTINET MM	7–2
<hr/>		
CHAPTER 8	UTILITIES	8–1
	PUNCH	8–2

Contents

INDEX

EXAMPLES

4-1	Sample <code>xgs.cmd</code> File _____	4-26
4-2	Sample <code>xgs.cmd</code> File _____	4-28
4-3	A Sample <code>snads_local_option</code> File _____	4-32
5-1	A Sample <code>cc:Mail</code> Channel Option File _____	5-2
5-2	A Sample Microsoft Mail Channel Option File _____	5-3
5-3	Sample MHS Channel Option File _____	5-3
5-4	A Sample GroupWise Channel Option File _____	5-3

FIGURES

4-1	Sample SNADS and SMTP Site _____	4-7
4-2	Defining SMTP Hosts to DISOSS _____	4-10
4-3	Defining the Route to the PMDF-XGS Transport Bridge on an Adjacent DISOSS Node _____	4-11
4-4	Existing Routing Table Entry on a DISOSS Node Not Adjacent to the PMDF-XGS Transport Bridge _____	4-11
4-5	New Routing Table Entry on a DISOSS Node Not Adjacent to the PMDF-XGS Transport Bridge _____	4-11
4-6	VTAM Definition of the PMDF-XGS Transport Bridge on an Adjacent DISOSS Node _____	4-12
4-7	CICS CONNECTION Definition _____	4-13
4-8	CICS SESSION Definition _____	4-14
4-9	Defining SMTP Hosts to OV/400 _____	4-15
4-10	Defining the Route to the OS/2 Transport Bridge on an Adjacent AS/400 Node _____	4-16
4-11	Existing Routing Table Entry on an AS/400 Node Not Adjacent to the PMDF-XGS Transport Bridge _____	4-17
4-12	New Routing Table Entry on an AS/400 Node Not Adjacent to the PMDF-XGS Transfer System _____	4-18
4-13	Defining the AS/400 Distribution Queue for the OS/2 Transfer System _____	4-19
4-14	The NDF File on the Transport Bridge System _____	4-23
4-15	Format of the <code>xgs.cmd</code> File for Multiple Send Processes on NT _____	4-44
4-16	Format of the <code>xgs.cmd</code> File for Multiple Send Processes on OS/2 _____	4-45

Preface

Purpose of This Manual

This manual describes older features of PMDF for OpenVMS that are obsolete as of PMDF V6.5, and which can be removed in a future version of PMDF. The intended audience is PMDF system managers who are already using these features and have not yet migrated to newer PMDF facilities. This manual is provided as a convenience for such existing usages.

Apart from the command line configuration utilities which exist on both UNIX and OpenVMS platforms, most of the contents of this manual are only relevant to PMDF on OpenVMS.

Overview of This Manual

This manual consists of the following chapters:

Chapter 1, Command Line Configuration Utilities, discusses some command line configuration utilities.

Chapter 2, Obsolete Channel Keywords, describes channel keywords used with BITNET channels.

Chapter 3, BITNET Channels (OpenVMS), describes the BITNET channels, (both RFC 822 and BSMTP) using either `Jnet` or `ANJE` in PMDF for OpenVMS.

Chapter 4, PMDF-XGS, describes how an e-mail connection can be established between systems using the SNADS (SNA Distribution Services) protocol and networks or mail systems reachable with PMDF.

Chapter 5, The MultiWare Client API Approach in PMDF-LAN (OpenVMS), describes using the MultiWare client API with `PMDF-LAN` channels.

Chapter 6, MultiNet SNMP Subagent for PMDF (OpenVMS), describes the MultiNet SNMP sub-agent for PMDF.

Chapter 7, MultiNet MM User Agent (OpenVMS), describes PMDF support for the MultiNet MM user agent.

Chapter 8, Utilities, describes the obsolete PMDF PUNCH utility.

Availability

PMDF software products are marketed directly to end users in North America, and either directly or through distributors in other parts of the world depending upon the location of the end user. Contact Process Software for ordering information, to include referral to an authorized distributor where applicable:

Process Software, LLC
959 Concord Street
Framingham, MA 01701 USA
+1 508 879 6994
+1 508 879 0042 (FAX)
pmdf_info@process.com

Preface

1 Command Line Configuration Utilities

This chapter discusses the command line versions of PMDF configuration utilities now obsolete by the web-based configuration utility.

On UNIX, the new web-based configuration utility has superceded the command line configuration utilities `pmdf configure mta`, `pmdf configure mailbox_servers`, `pmdf configure lan`, and `pmdf configure xgs`.

On OpenVMS, the new web-based configuration utility has superceded the command line configuration utilities `PMDF CONFIGURE MTA`, `PMDF CONFIGURE MAILBOX_SERVERS`, `PMDF CONFIGURE LAN`, and `PMDF CONFIGURE XGS`.

2 Obsolete Channel Keywords

This chapter describes obsolete PMDF configuration features, such as obsolete channel keywords.

2.0.1 Gateway daemons (`daemon`)

The interpretation and usage of the `daemon` keyword depends upon the type of channel to which it is applied. In addition to the usages discussed in the *PMDF System Manager's Guide*, the keyword also has the following special meaning for BITNET channels.

When a channel is used to provide a connection to a gateway system on BITNET, the username associated with the mail handling daemon on the gateway system must be made known to the `Jnet` and `ANJE` master programs. Unfortunately, no standard exists for the names of these daemons; each gateway system uses a name of its own choosing.

The `daemon` keyword is used to tell the `Jnet` and `ANJE` master programs the name of the gateway daemon to which to send the message. The actual daemon name should appear directly after the `daemon` keyword. For example, here is a channel block for a typical BITNET gateway system:

```
bit_interbit smtp daemon smtp user mailer
INTERBIT.BITNET
```

2.0.2 Multiple Gateways on a Single Channel (`multigate`, `nomultigate`)

The interpretation of the `multigate` channel keyword depends upon the type of channel to which it is applied. In addition to the usages discussed in the *PMDF System Manager's Guide*, the keyword also has the following special meaning for BITNET channels.

Systems attached to BITNET can use a separate channel to access each individual BITNET gateway and each BITNET system that accepts messages in batch SMTP (BSMTP) format. This can lead to substantial overhead in the PMDF configuration file. The `multigate` keyword allows gateways and BSMTP systems with similar characteristics to be serviced by a single channel. There is still some overhead because each gateway system and BSMTP system needs a separate entry in the channel block, but this is much less than having multiple channels. See Section 3.8.3 for further details on the use of this keyword.

Obsolete Channel Keywords

The `multigate` keyword tells PMDF to route the message to the daemon mailbox specified by the `daemon` keyword (described in Section 2.0.1) on the system specified in the message's `To:` address. This differs from PMDF's behavior when the `multigate` keyword is not used, in which case PMDF routes the message to the official host associated with the channel, *not* the system specified in the message's `To:` address.

There are a variety of caveats associated with using the `multigate` keyword; see the documentation on BITNET gateways in Section 3.7 for additional information. `nomultigate` is the default.

2.1 Batch SMTP Continuation Lines (`contchar`, `contposition`)

The `contposition` keyword specifies the point at which batch SMTP lines are folded onto a continuation line. The default value for `contposition` is 0, which disables continuation lines.

The `contchar` keyword is used to specify a continuation character for commands in batch SMTP. Commands longer than `contposition` characters long will have the character specified by the integer argument to `contchar` inserted in the `contposition` position and the remainder of the line will be wrapped to another line. The default value for `contchar` is 0.

2.2 Using Local Usernames as User Tags (`localuser`, `nolocaluser`)

When someone on the local system sends a message that PMDF routes via Jnet or ANJE, that user's username is used in the user part of the message tag. NJE status and log messages produced as the message traverses the network will then be sent back to the originating user. This can be annoying, and some sites can want to suppress these messages.

The `nolocaluser` keyword, when specified in a Jnet or ANJE channel block, inhibits the use of usernames as the user tag. The name used in the tag is then controlled with the `user` keyword, described in the *PMDF System Manager's Guide*. The `localuser` keyword enables this usage and is the default.

The `nolocaluser` keyword has no effect on non-Jnet or ANJE channels and also has no effect on messages that originated on a system other than the local system.

Note: Using the `nolocaluser` keyword can have side effects. Some systems object to messages whose user tag does not match the sender's address. In particular, use of the `nolocaluser` keyword on BITNET systems which have not registered their mailer name is *not* recommended; such systems are considered untrustworthy in the BITNET community.

Obsolete Channel Keywords

Generation of TICKet BSMTP Commands (`tick`, `notick`)

2.3 Generation of TICKet BSMTP Commands (`tick`, `notick`)

Some batch SMTP (BSMTP) implementations require the presence of a ticket number, specified with the `TICK` BSMTP command. The `tick` keyword tells PMDF to issue this command; `notick` suppresses it. `tick` is the default on channels that support tickets. Currently only the `BN_MASTER` channel program uses this keyword. See Section 3.8.2 for details.

2.3.1 Originating User Tag (`user`)

In addition to the usages of the `user` channel keyword discussed in the *PMDF System Manager's Guide*, this keyword also has a special meaning for `BITNET` channels.

Messages sent via `BITNET` are tagged with the name of the user and system that sent the message. If someone on the local system sent the message, PMDF uses that user's username as the user part of the tag. The following section explains how this can be inhibited if desired. PMDF uses `MAILER` as the default user part of the tag for messages originating on remote systems. This works correctly in most cases, but some `NJE` mail systems (notably many of those on `BITNET`) are somewhat paranoid and check tag information to make sure the message was relayed by a "trusted" user on a "trusted" mail system. The name of this special, trusted user is usually, but not always, the same as the name of the special mailbox associated with any local `Jnet` or `ANJE` gateway that has been set up.

Trusted mail systems typically are registered with some type of central network administration. PMDF itself does not provide facilities for restricting access to trusted systems. However, PMDF does provide the means to set the username tag to a value that is acceptable to remote mailers.

The `user` keyword is used to tell the `Jnet` and `ANJE` master programs to use an alternate username in the tag field. The actual username to use in the tag should appear directly after the `user` keyword.

If no `user` keyword appears on the channel associated with an outgoing message, PMDF then checks to see if the `user` keyword has been specified on the local channel, and if it has, uses the value associated with that specification. This makes it possible to simplify some configurations that have `user` keywords on many or even all of the `bit_` channels.

The example channel block shown in Section 2.0.1 uses the `user` keyword to set the username associated with the `:mailer. tag` entry in the node's `BITNIC` node registration.

Obsolete Channel Keywords

Generation of VERBose BSMTP Commands (`verb_on`, `verb_off`, `verb_none`, `verb_never`)

2.4 Generation of VERBose BSMTP Commands (`verb_on`, `verb_off`, `verb_none`, `verb_never`)

Some batch SMTP (BSMTP) implementations support the use of the `VERB` command to control the nature of their replies. On the client side the `verb_on` command tells PMDF to issue a `VERB ON` command in the BSMTP command sequence; `verb_off` tells PMDF to issue a `VERB OFF` command. The `verb_never` and/or the `verb_none` keyword tells PMDF not to issue any `VERB` commands. `verb_never` is the default, and this default should *not* be changed.

On the server side, the `verb_never` keyword causes `VERB` commands in the BSMTP stream to be accepted but ignored. `verb_none` can be used to enable processing of `VERB` commands.

See Section 3.8.2 for details on how these keywords are used on `bit_` and `anje_` channels.

3 BITNET Channels (OpenVMS)

Note: BITNET channels were relegated to obsolete status as of PMDF V5.2. Completely untested and unsupported BITNET channel images have been built for PMDF and included in the distribution. Sites can try using them, but Process Software does not warrant that they will work or work correctly.

Jnet and ANJE channels are used to link PMDF to either Jnet or ANJE.¹ Both of these software packages implement NJE compliant networking under OpenVMS. NJE networking is used to link OpenVMS systems to the national BITNET network as well as its sister networks internationally. The documentation presented here is intentionally biased towards sites on BITNET; sites on NJE networks disjoint from BITNET will have to ignore all the BITNET-specific information contained in this chapter and devise their own procedures for configuring their networks.

Jnet and ANJE channels use a single unidirectional master program and a single unidirectional slave program. The master program can only send while the slave program can only receive. There is also a third channel program for use with Jnet and ANJE; this channel is used to implement mailer processing and is not itself directly attached to Jnet or ANJE.

The slave program runs either as Jnet's Local Mail Delivery (LMD) agent or as the ANJE delivery daemon and is only associated with one channel (the "local NJE" channel). The master program is invoked by MASTER.COM for all Jnet and ANJE channels.

Jnet and ANJE channels are generated by the automatic configuration generator if any are needed.

Note: PMDF is compatible with versions 3.2, 3.3, 3.4, 3.5, 3.6, and 3.7 of Jnet and version 5 of ANJE. Other versions of these packages can require alterations to the PMDF interface.

Note: The command file lmd.com can need to be adapted to different versions of Jnet, and the username changing routine USERNAME used by ANJE master channels can need to be relinked since it is linked against the system symbol table SYS.STB. Use the command procedure PMDF_COM:link_username.com to relink this image.

3.1 The NJE Environment

NJE provides a store and forward file relaying service. Hosts are registered in a flat name space—all users have names of eight characters or less and all hosts (systems) have simple names of eight characters or less. No provisions are provided for accessing hosts outside the NJE name space. All routing is implicit; each system knows how to reach (via a routing table) all the other systems on the network.

¹ Jnet is a commercial product available from Wingra, Inc. ANJE is available from the National Energy Software Center.

BITNET Channels (OpenVMS)

The NJE Environment

Mail messages usually correspond to class `M NJE` files. Files in this class are automatically processed as mail messages and not as regular files. PMDF is not normally concerned with other classes of `NJE` files; the selection of what classes of files PMDF processes is made by the `NJE` service and not by PMDF itself.

3.2 The BITNET Environment

BITNET is an example of a large `NJE` network. OpenVMS systems on BITNET use `Jnet` or `ANJE`; other types of systems on BITNET use other implementations of `NJE`. Since `NJE` has no provisions for reaching systems outside the closed `NJE` network, certain systems on BITNET have been set up to act as gateways to various outside networks. A mail message to a non-BITNET system is typically sent as a specially formatted file which is delivered to a special user on one of the gateway systems. This special user is called a “mailer” in BITNET terminology. Although all gateways on BITNET are implemented as mailers the converse is not true—many sites run mailers that do not provide a gateway to anything other than their own system.

When a message is delivered to a mailer gateway it is interpreted as mail bound for some outside system and is transmitted to the appropriate outside systems using some non-BITNET mechanism. Gateways exist on BITNET that provide access to the Internet and to several other large networks, as well as numerous small isolated networks.

Most BITNET mailer gateways use a variant of RFC 821 (Simple Mail Transfer Protocol or “SMTP”) adapted to batch mode operation. This variant is called batch SMTP (BSMTP) and is described in the document *Batch Simple Mail Transfer Protocol* by Alan Crosswell. A copy of this document is stored in the `RFCs` subdirectory of PMDF’s documentation directory, *i.e.*, `PMDF_DOC:[rfc]rfc821.txt` on OpenVMS. A few gateway systems simply examine the RFC 822 message header present on all messages to obtain the proper recipient addresses; (this approach is fraught with peril due to the complex nature of RFC 822 headers, and is almost obsolete).

The difficulty in using mailers and mailer gateways is that the end user must have an understanding of what amounts to a very complex forwarding scheme. Several utilities have in fact been developed to deal with this situation, including the `SENDGATE` command file written by Ed Miller and the `GATEWAY VMS MAIL` interface written by John Carosso. The PMDF interface to `Jnet` is based on John Carosso’s `GATEWAY` code and provides comparable facilities for routing messages to the appropriate mailers automatically.

3.3 Setting Up a Local Jnet Channel

The first step in configuring `Jnet` to work with PMDF is to install PMDF and `Jnet` on the same system. Once this is done a local `Jnet` channel should be set up. This is described below.

Note: The `PMDF CONFIGURE` utility can be used to configure your system with the appropriate BITNET channels automatically. When the PMDF configuration utility is used, all of the necessary configuration steps except for those described in Section 3.11.1 are done

BITNET Channels (OpenVMS) Setting Up a Local Jnet Channel

for you automatically. Consult the OpenVMS edition of the *PMDF Installation Guide* for instructions on using the configuration utility. *Process Software strongly recommends that you use the supplied* PMDF CONFIGURE *utility.*

3.3.1 Adding the Channel to the Configuration File

A channel should be added to the configuration file to service local Jnet traffic. This channel is named `bit_local` and the use of this channel name is *mandatory*. Add a channel block to your configuration file of the form:

```
bit_local 733 single nosmtp
Jnet-DAEMON
```

The `single` keyword tells PMDF that only a single address is allowed in each message file (a limitation of NJE) and the `nosmtp` keyword tells the BN_MASTER program that an SMTP header should not be prepended to the message. (SMTP headers are needed when BN_MASTER sends messages to BITNET mailers. These headers are not necessary and should not be used on messages local to NJE.) The `733` tells PMDF to use RFC 733 (percent sign) addressing conventions since NJE does not support RFC 822 addressing.

3.3.2 Channel Naming Conventions

All Jnet channel names begin with `bit_`, and the name associated with the local Jnet host (*i.e.*, the name on the second line of the channel block) is `Jnet-DAEMON`. In contrast, ANJE channel names begin with `anje_` and the name associated with the local ANJE host is `ANJE-DAEMON`. Many of the following sections of this documentation apply to both Jnet and ANJE; if you are using Jnet, you should replace any ANJE specific channel and host names with their Jnet equivalents.

3.4 Setting Up a Local ANJE Channel

The first step in attaching ANJE to PMDF is to install PMDF and ANJE on the same system. Once this is done a local ANJE channel should be set up. This procedure is basically the same as the procedure used when setting up a connection to Jnet; only the differences are described here.

Note: The PMDF CONFIGURE utility can be used to automatically configure your system with the appropriate BITNET channels. When the PMDF configuration utility is used, all of the necessary configuration steps are automatically done for you. Consult the OpenVMS edition of the *PMDF Installation Guide* for instructions on using the configuration utility. *Process Software strongly recommends that you use the supplied* PMDF CONFIGURE *utility.*

BITNET Channels (OpenVMS) Setting Up a Local ANJE Channel

3.4.1 Relinking the USERNAME Image

When using ANJE, PMDF's BN_MASTER program activates the `username.exe` image to change usernames while sending. `username.exe` is linked against the system symbol table `SYS.STB`, which can not match the version on your system. For this reason you can want to relink `username.exe` prior to using it.²

If you want to relink the `username.exe` image, issue the OpenVMS command

```
$ @PMDF_COM:link_username.com
```

3.4.2 Adding the Channel to the Configuration File

A channel should be added to the configuration file to service local ANJE traffic. This channel is called `anje_local` and the use of this channel name is *mandatory*. Add a channel block to your configuration file of the form:

```
anje_local 733 single nosmtp
ANJE-DAEMON
```

This definition corresponds to the Jnet `bit_local` channel definition, described in detail in Section 3.3.1.

3.4.3 Channel Naming Conventions

All ANJE channel names begin with `anje_`, and the name associated with the local ANJE host (*i.e.*, the name on the second line of the channel block) is `ANJE-DAEMON`. In contrast, Jnet channel names begin with `bit_` and the local Jnet host name is `Jnet-DAEMON`. Most of the following sections of this documentation apply to both ANJE and Jnet; if you are using ANJE, you should replace any Jnet-specific channel and host names with their ANJE equivalents.

3.5 Using an Option File to Customize NJE Channels

A channel-specific option file can be used to provide additional control information to the channel. This information is used when the message is formatted and transmitted.

Note: Option files are intended for special customization purposes only. Unlike PhoneNet option files, they are *not* required for normal operation. They should *not* be used under normal circumstances. If you are not an expert at PMDF configuration, you should not worry about BITNET channel option files and you should skip this section.

² The `username.exe` image is relinked automatically for you each time you install or upgrade PMDF.

BITNET Channels (OpenVMS) Using an Option File to Customize NJE Channels

Option files are stored in the PMDF table directory (*i.e.*, `PMDF_TABLE:` on OpenVMS) and have names of the form `x_option.`, `x` is the name of the BITNET channel to which the option file applies. Each BITNET channel can have its own option file.

Option files consist of several lines. Each line contains the setting for one option. An option setting has the form:

option=value

value can be either a string or an integer, depending on the option's requirements. If the option accepts an integer value, a base can be specified using notation of the form `b%v`, `b` is the base expressed in base 10 and `v` is the actual value expressed in base `b`.

Comments are allowed. Any line that begins with an exclamation point is considered to be a comment and is ignored. Blank lines are also ignored in any option file.

The available options are:

FORMAT (NETDATA, PRINTER, PUNCH)

Controls the class of formatting used when transmitting messages. Valid values are `PUNCH`, `PRINTER`, and `NETDATA`. `PUNCH` is the default and provides records up to 80 characters long. `PRINTER` provides 132 character records, and `NETDATA` even longer records. Note, however, that not all BITNET mail receivers can accept all these different formats. This option is equivalent to the Jnet `SEND` command qualifiers `/PUNCH`, `/PRINTER`, and `/NETDATA`. This option is not supported by ANJE.

CLASS (string)

This option controls the message class. The default class for messages is `M`. Other classes can be used, but `M` is the standard class for mail messages. In particular, Jnet and ANJE both process only class `M` messages as mail.

EXPAND_TABS (0, 1)

This option controls the expansion of tabs in the message body. A value of 1 (the default) indicates that tabs are to be expanded to spaces, with tab stops every 8 characters. A value of 0 inhibits expansion, leaving the handling of tabs up to the transport agent (Jnet or ANJE).

OPTIONS (string)

This option controls the setting of the options string associated with the NJE envelope. An empty options string is generated by default. This option is not supported by ANJE.

LINE_LENGTH (integer > 0)

This option controls the line length limit imposed on lines in message bodies. Lines longer than this limit are broken up into multiple lines, and subsequent lines are indented. For proper operation of BSMTP, this length limit should be less than or equal to the overall `RECORDSIZE` set below.

RECORDSIZE (integer > 0)

Establishes a maximum record size for all messages. This value cannot be greater than the limits associated with the various formats described above. Longer records will be broken into multiple records by the file transmission software. This option corresponds to Jnet's `/RECORDSIZE` qualifier, and is not supported by ANJE. There is no default limit.

BITNET Channels (OpenVMS)

Defining the Systems Accessible via NJE

3.6 Defining the Systems Accessible via NJE

The names of the systems accessible via NJE can be added to the channel block, or rewrite rules can be added to the configuration file or domain database to map the system names onto the `bit_local` or `anje_local` channel. Since NJE only uses eight character system names that cannot contain special characters, the `BN_MASTER` program automatically strips domain specifications (e.g., `.BITNET`) from the system names in envelope addresses. Thus the domain specification in the pattern should be chosen for its appropriateness rather than for consideration by `BN_MASTER`.

If the number of rewrite rules is large (as in the case of BITNET) then the use of a domain database is recommended.³ It is also possible to avoid having to list all possible NJE systems by defining a domain to contain all the systems. For example, the rewrite rule

```
.BITNET          $U%$H.BITNET@Jnet-DAEMON      (for Jnet)
```

for Jnet, or

```
.BITNET          $U%$H.BITNET@ANJE-DAEMON      (for ANJE)
```

for ANJE defines the `.BITNET` domain to contain all Jnet systems. There are three disadvantages to this scheme for dealing with BITNET: (1) all system names now have a domain name part that must be entered by the user, (2) references to unknown or illegal systems will not be detected in a timely manner, and (3) BITNET mailer gateway usage will be incorrect. Some sites prefer to use a mailer rather than use regular class M mail messages since it avoids so many problems associated with the restrictions on address lengths. For these three reasons, if you are on BITNET the use of the `bitnet_domains_driver.com` procedure described in Section 3.8 is strongly recommended. This file is automatically created for you by the `PMDF CONFIGURE` utility.

3.7 Setting Up Access to BITNET Mailer Gateways

Access to BITNET mailer gateways is provided by defining additional channels, one per mailer, in the configuration file. BITNET mailer gateway channel names must always begin with `bit_` for Jnet channels or with `anje_` for ANJE channels. The channel blocks have the general form:

```
bit_interbit smtp daemon smtp user mailer
INTERBIT.BITNET
```

`INTERBIT` is a generic name for a local BITNET mailer gateway that provides access to the Internet. It should be one of the many Internet gateways on BITNET. Most gateways use SMTP, so they are marked `smtp`. The `daemon` keyword's argument specifies the username that is used as the intermediate destination on the gateway. The optional `user` keyword specifies the username that appears in the NJE origin tag field of the message header. Some mailer gateways insist on having the origin set to the name of a "trusted" user. Other gateways (such as `INTERBIT`) will send failed message notices to the user whose name appears in the origin tag field.

³ This approach is used in the configuration generated by the `PMDF CONFIGURE` utility.

BITNET Channels (OpenVMS) Setting Up Access to BITNET Mailer Gateways

Mailers that check the origin tag field check both the user and system parts of the tag. The `user` keyword can be used to set the user part of the tag, but the system name in the tag must also be registered. This is not normally a problem since the registered system name is what `Jnet` or `ANJE` will use. However, in a VMS cluster configuration it can be convenient to register the cluster alias as the system name that must appear in the tag. `Jnet` will use the cluster alias in the tag for messages sent by PMDF if the following logical name definition is made:

```
$ DEFINE/SYSTEM/EXEC JAN_BN_MASTER_FLAGS 1
```

The use of such `JAN_` logical names is fully described in the `Jnet` documentation.

Warning: Setting this logical to 1 on a system that does not have a cluster alias will cause `BN_MASTER` to fail with a `SENDFILE-F-MYNODE` error. Be careful not to copy this logical setting from a `Jnet` system with a cluster alias to a system that does not have a cluster alias!

Versions of PMDF prior to version 3.0 actually invoked the `Jnet SENDFILE` program and thus their use of the cluster alias was controlled by the `JAN_SENDFILE_FLAGS` logical name. This changed with PMDF 3.0 when support was added to call the file sending routines in `JANSHR` directly.

The flags mechanism is not available with `ANJE`.

Certain gateways can restrict the number of addresses that can appear in a single copy of a message so it can be appropriate to add the `single` or `single_sys` keywords to the channel block for some gateway channels.

Once a channel block for a gateway is created it is ready to use; there are no log files to create.

3.8 BITNET Gateway Tables and `bitnet_domains_driver.com`

There are many BITNET mailer gateways and more are added every month. A list of all registered mailer gateways is maintained by the BITNET Network Information Center. A current copy of the list can be obtained on BITNET systems by using the `Jnet` command:

```
$ SEND LISTSERV@BITNIC SEND DOMAIN NAMES
```

In addition to the actual gateways, lots of BITNET sites operate a mailer that accepts mail in BSMTP format, even if they do not operate a mailer gateway. Information about what form of mail (regular `NJE` or BSMTP) each site prefers is listed in the `xmailer` names file. You can obtain a current copy of this file with the command:

```
$ SEND LISTSERV@PUCC SEND XMAILER NAMES
```

Note that `XMAILER NAMES` is also available from `BITNIC`, but the `PUCC` version is somewhat enhanced from the `BITNIC` version.

BITNET Channels (OpenVMS)

BITNET Gateway Tables and `bitnet_domains_driver.com`

Copies of both of these files are part of the PMDF for OpenVMS distribution — the distribution copies are kept in the PMDF table directory, (*i.e.*, `PMDF_TABLE:domains.names` and `PMDF_TABLE:xmailer.names`). However, these files rapidly become dated and it is always preferable to work with current copies if possible.

A DCL procedure is provided with PMDF for OpenVMS that will convert the `DOMAIN NAMES` and `XMAILER NAMES` files into a format that can be inserted directly into the configuration file. This procedure, `PMDF_COM:bitnet_domains.com`, reads these two files and produces the following output files:

gates.rules

Rewrite rules for BSMTP systems suitable for insertion into either the configuration file or the domain database.

gates.chans

Channel definitions for the configuration file that are used by the `gates.rules` rewrite rules.

bitnet.rules

Rewrite rules for regular NJE systems suitable for conversion into a domain database.

tcpip.rules

Rewrite rules for BITNET systems and domains that are reachable via TCP/IP instead of BITNET. This file only contains meaningful output if the `ON_INTERNET` flag is set to 1 (see below).

gateway.rules

A set of rewrite rules for BSMTP systems suitable for insertion in configuration files on remote systems that are going to use a central gateway.

bgateway.rules

A set of rewrite rules for all regular NJE BITNET systems suitable for insertion in configuration files on remote systems that are going to use a central gateway.

domain.dat

A domain database containing the same rules `bitnet.rules` contains. This is optional and can be suppressed; see below.

When you execute the `PMDF CONFIGURE` utility and configure your system for BITNET, a `bitnet_domains_driver.com` file will be created in the PMDF table directory, `PMDF_TABLE:.` The commands in this file set various DCL symbols which set site specific parameters and then execute the `bitnet_domains.com` procedure located in the PMDF com directory, `PMDF_COM:.` These DCL symbols, as well as others not set by the PMDF configuration utility, are described below. If you want to set or change one of these symbols, make the correction in your `bitnet_domains_driver.com` command file rather than in `bitnet_domains.com`, which is replaced whenever PMDF is upgraded. If, for some reason, you choose not to run the PMDF configuration utility, then copy `bitnet_domains_driver.template` to `bitnet_domains_driver.com` and edit that. *Process Software strongly recommends that you use the supplied PMDF CONFIGURE utility.*

1. `ON_INTERNET` should be set to 1 if your system is actually on the Internet. If your system is not on the Internet `ON_INTERNET` *must* be set to 0. For example, if you are on the Internet, use the following declaration

BITNET Channels (OpenVMS) BITNET Gateway Tables and bitnet_domains_driver.com

```
$ ON_INTERNET = 1
```

- HANDLES_MX should be set to 1 if your system's TCP/IP implementation and configuration can handle MX records. HANDLES_MX should be set to 0 otherwise. This setting is only meaningful if ON_INTERNET is 1. For example, to declare that your configuration can handle MX records, use the declaration

```
$ HANDLES_MX = 1
```

- TCP_DAEMON should be equated to the name of the TCP/IP daemon you use; *e.g.*,

```
$ TCP_DAEMON = "TCP-DAEMON"
```

- INTERNET_GATEWAY should be set to 1 if your system is going to act as a gateway from BITNET to the Internet. This should be set even if your site does not intentionally gateway messages from BITNET to the Internet but does have the capacity to do so. If this capability does not exist INTERNET_GATEWAY should be set to 0. For example, if you operate a gateway from BITNET to the Internet, use the following declaration

```
$ INTERNET_GATEWAY = 1
```

- If INTERNET_GATEWAY is set to 1, you *must* set INTERNET_GATEWAY_HOST to the name of the Internet host with which BITNET addresses emitted onto the Internet will be qualified. This is usually the TCP/IP name for your local host name, but not always. More generally, it is the host that will be used to gateway replies from the Internet back onto BITNET. For example, if *naples.example.com* is the name of the Internet to BITNET gateway you want to use, use the following declaration

```
$ INTERNET_GATEWAY_HOST = "naples.example.com"
```

- If you set INTERNET_GATEWAY to 1, set INTERNET_CHANNELS to be a list of the channels that connect to the Internet. This list *must not* include channels that connect to the Internet via BITNET. If set, this parameter should translate to a list of channel names separated by /. Do *not* begin or end the list with / as you do for some of the symbols below.

- SKIP_DOMAIN should be set to the list of domains whose BITNET gateways you never want to use. You should list your own domain if you are the BITNET gateway for it, along with any domains you want to avoid (for whatever reason). Begin each domain name with a backslash, and terminate the entire list with a trailing backslash; *e.g.*,

```
$ SKIP_DOMAIN = "/.EXAMPLE.COM/.naples.com/"
```

- SKIP_XMAILER should be set to the list of systems for which you already have rewrite rules (and hence for which you don't need any more). This would probably include all BITNET systems that are part of your local VMS cluster, for example. Separate each system name with a backslash and terminate the entire list with a trailing backslash; *e.g.*,

```
$ SKIP_XMAILER = "CBROWN/ECHMC/HMCVAX/SIF/YMIR/"
```

Do not include any pseudo-domains in the system names (*e.g.*, .BITNET). The system names should be as they appear in the XMAILER NAMES file.

- FORCE_BITNET can be set to a list of domain names that you want routed via BITNET, regardless of their TCP/IP connectivity status. Begin each domain name with a backslash and terminate the list with a trailing backslash; *e.g.*,

```
$ FORCE_BITNET = "/.EXAMPLE.COM/NAPLES.COM/"
```

BITNET Channels (OpenVMS)

BITNET Gateway Tables and `bitnet_domains_driver.com`

10. `FORCE_INTERNET` can be set to a list of domain names that you want routed via the Internet regardless of what their entries in `DOMAIN NAMES` say. Begin each domain name with a backslash and terminate the list with a trailing backslash; *e.g.*,

```
$ FORCE_INTERNET = "/.SRI.NIC/.EXAMPLE.COM/"
```

11. `FORCE_INTERBIT` can be set to a list of domain names that you want routed to the INTERBIT gateway regardless of what their entries in `DOMAIN NAMES` say. Begin each domain name with a backslash and terminate the list with a trailing backslash; *e.g.*,

```
$ FORCE_INTERBIT = "/.EXAMPLE.com/.NAPLES.COM/"
```

12. `BITNET_ALIAS` should be set to your BITNET system name if it is not the same as your official host name; *e.g.*,

```
$ BITNET_ALIAS = "NAPLES.BITNET"
```

The specification of the trailing pseudo-domain `.BITNET` is recommended.

13. `BITNET_QUEUE` should be set to the name of the batch queue used to process jobs handling the various BITNET channels. This is typically a queue that runs on a system where Jnet or ANJE software is installed. If no special queue is specified these jobs will run in the normal PMDF batch queue or queues.

14. `BITNET_CHANNEL_TYPE` should be set to `bit_` for Jnet channels or to `anje_` for ANJE channels; *e.g.*,

```
$ BITNET_CHANNEL_TYPE = "bit_"
```

15. Normally BITNET shortform names can be used anywhere in PMDF; they are automatically converted into their `.BITNET` form. This can present problems if these names conflict with local shortform names, especially if these names are handled by default rules.

The `BITNET_LOCAL_CHANNEL` parameter provides a mechanism for selectively disabling the ability to use BITNET shortform names. Normally this parameter is blank. If set, this parameter should translate to a list of the channel names, separated by `/`, where BITNET shortform names are to be considered to be valid. If any list is specified, it must include your local BITNET channel; *e.g.*, `"bit_local"` for Jnet or `anje_local` for ANJE. It probably should also include your BITNET gateway channel `bit_gateway` as well, if you have one. Do *not* begin or end the list with `/` as you do for some of the other lists in `bitnet_domains_driver.com`.

16. `PERIOD` controls the periodicity of service the channels generated by the procedure `bitnet_domains.com` receive from the periodic delivery batch job. Since BITNET gateway channels are serviced immediately by default, delivery failures are rare, and there are a lot of gateway channels (which means checking them all is time consuming), the period is usually set to 3, which means that only every third periodic delivery batch job will check these queues. To declare a period of 2, use the declaration

```
$ PERIOD = 2
```

17. `DO_CRDB_BR` should be set to 1 if you want to have CRDB automatically run on `bitnet.rules` to produce a `domain.dat` file:

```
$ DO_CRDB_BR = 1
```

You can set both `DO_CRDB_BR` and `DO_CRDB_GR` if you want `bitnet.rules` and `gates.rules` to be combined in a single `domain.dat` file.

BITNET Channels (OpenVMS) BITNET Gateway Tables and bitnet_domains_driver.com

18. DO_CRDB_GR should be set to 1 if you want to have CRDB automatically run on gates.rules to produce a domain.dat file:

```
$ DO_CRDB_GR = 1
```

You can set both DO_CRDB_BR and DO_CRDB_GR if you want bitnet.rules and gates.rules to be combined in a single domain.dat file.

19. GATEWAY_USER_NAME should be set to the user tag your gateway uses. If you do not operate a BITNET gateway GATEWAY_USER_NAME should be set to either MAILER or SMTPUSER; e.g.,

```
$ GATEWAY_USER_NAME = "MAILER"
```

20. GATEWAY_SYSTEM_NAME should be set to the name of your BITNET gateway system; e.g.,

```
$ GATEWAY_SYSTEM_NAME = "ymir.claremont.edu"
```

If you do not operate a BITNET gateway, GATEWAY_SYSTEM_NAME should be set to the name of the local system that is running PMDF.

21. CHANNEL_KEYWORD_OPTIONS can be set to any additional channel keywords (or rightslist identifiers) which should be applied to every bit_channel; e.g.,

```
$ CHANNEL_KEYWORD_OPTIONS = "notices 3 6 9 12"
```

Note that in most, if not all, cases a defaults channel can be used to apply keywords to all bit_channels.

Once you have up-to-date copies of DOMAIN NAMES and XMAILER NAMES and have made any desired edits to bitnet_domains_driver.com, you can run the procedure as follows:

```
$ SET DEFAULT PMDF_TABLE:  
$ @bitnet_domains_driver.com
```

The resulting gates.chans file should be included at the end of your PMDF configuration file, pmdf.cnf, by using the include command *<file-spec>*, i.e.,

```
<PMDF_TABLE:gates.chans
```

This file is included automatically when you use the PMDF for OpenVMS configuration utility to generate the configuration.

The rewrite rules in gates.rules can either be added to the domain database domain.dat using CRDB or they can be included by the PMDF configuration file (again, using the include command *<file-spec>*). If you set ON_INTERNET to 1, the file tcpip.rules should be included by your configuration as well. It is automatically included if you use the PMDF configuration utility to generate your configurations.

The files gateway.rules and bgateway.rules are *not* intended for use on the gateway system. They are designed to be added to the rewrite rules on a remote system running PMDF that wants to use as a gateway the gateway system that produced gateway.rules

BITNET Channels (OpenVMS)

BITNET Gateway Tables and bitnet_domains_driver.com

3.8.1 Operating as a Trusted Mailer on BITNET

Some BITNET mailers check the origin tag field on messages to make sure it matches the `From:` address in the BSMTTP envelope. This is not normally a problem when the message originated on the local system, but it is a problem when the message originated elsewhere and is being gatewayed onto BITNET by the local system. In this case there's no way that the `From:` address can match; it specifies a system that is not directly on BITNET.

The solution to this problem is to operate PMDF as a trusted mailer. BITNET mailer gatewayes exempt such mailers from this check.

You must set up and register a local mailer before this will work. See section 3.13.3 for information on how to register your local mailer and change the value of this information.

Your gatewayed messages must also have the proper origin tag. The system part of the tag is set by `Jnet` or `ANJE`, and is correct by default. The user part of the tag is controlled by the `user` keyword in the channel definitions. This must match the mailer address that you have registered in the BITNET tables.

3.8.2 Using the `verb_on` and `tick` Keywords

It can be appropriate to use the `tick`, `notick`, `verb_on`, or `verb_off` keywords on certain BITNET channels. However, there is no information about the use of these keywords in either `DOMAIN NAMES` or `XMAILER NAMES` and `bitnet_domains.com` never generates any of these keywords. Please refer to *Batch Simple Mail Transfer Protocol*, by Alan Crosswell (see the file `PMDF_ROOT:[doc]bsmtp.`) for additional information on these keywords and the corresponding BSMTTP commands they generate.

3.8.3 Using the Multigate Keyword

If you set up your own BSMTTP routing without using the command procedure `bitnet_domains_driver.com`, you will find that the number of channels needed rapidly gets out of hand. The `multigate` keyword allows systems that accept BSMTTP format messages with similar characteristics to be serviced by a single channel. There is still some overhead because each mailer needs a separate entry in the channel block, but the overhead is much less than when separate channels are used.

In order to use the `multigate` keyword, systems that accept BSMTTP format messages must be grouped according to their use of the `daemon` and `user` keywords. Only mailers with identical `daemon` and `user` specifications can be grouped on a single channel.

BITNET Channels (OpenVMS) BITNET Gateway Tables and bitnet_domains_driver.com

For example, suppose that the mailer gateways for the .NAPLES.COM and .MILAN.COM subdomains as well as the mailer for the system EXAMPLE.BITNET share the same characteristics — they all accept BSMTP format messages sent to the special user tag SMTPUSER from the special user tag SMTPUSER. They could be grouped in a single channel as follows:

```
.EXAMPLE.COM          $U%$H$D@EXAMPLE.BITNET
.MILAN.COM            $U%$H$D@SEMAX51.BITNET
EXAMPLE.BITNET       $U%$D@BIT-GATE-1

bit_gate1 smtp daemon smtpuser user smtpuser multigate single_sys
BIT-GATE-1
ARIZONA.BITNET ARIZONA.BITNET
SEMAX51.BITNET SEMAX51.BITNET
```

The presence of the `single_sys` keyword is required — a separate copy of each message is needed for each of the gateways.

Using this technique can substantially reduce the size of the PMDF configuration file. However, determining which gateways support the necessary RFC 822 addressing conventions is not trivial, nor is the task of organizing the various gateways in the proper manner. As a result, the use of `multigate` is best left to automated procedures like `bitnet_domains_driver.com`.

3.9 Maintenance of PMDF's BITNET Routing Tables

Every BITNET site should update their PMDF BITNET routing tables on a monthly basis by obtaining current versions of `XMAILER NAMES` and `DOMAIN NAMES`. If you do not already have a source for these files, then refer to Section 3.8 for one set of sources.

After obtaining current versions of these two files, place them in the location indicated in your `PMDF_TABLE:bitnet_domains_driver.com` file. Most sites keep these files as `PMDF_TABLE:xmailer.names` as `PMDF_TABLE:domain.names`, respectively. Then, issue the command

```
$ @PMDF_TABLE:bitnet_domains_driver.com
```

This command procedure will take several minutes to run; it will output new PMDF configuration information based upon the contents of the `XMAILER NAMES` and `DOMAIN NAMES` files. Next, if you are using a compiled configuration, issue the commands

```
$ PMDF CNBUILD
$ INSTALL REPLACE PMDF_CONFIG_DATA
```

The `CNBUILD` command must be issued once for each architecture (VAX, Alpha, and I64). The `INSTALL` command must be issued on every system running PMDF and using the same configuration information, (e.g., all members of a cluster running PMDF). Finally, issue the command

```
$ PMDF RESTART BN_SLAVE
```

to restart any detached Mailer Daemon processes running `BN_SLAVE`. This command requires the `SYSLCK` privilege to issue; it will restart all `BN_SLAVE` processes across a

BITNET Channels (OpenVMS)

Maintenance of PMDF's BITNET Routing Tables

cluster. (Each cluster member running PMDF and Jnet will have a BN_SLAVE process running.)

3.10 Suppressing Progress Messages

As a mail message traverses BITNET, progress messages are sent back to the message's originator and broadcast on their terminal. These messages can be suppressed by applying the `nolocaluser` keyword to the `bit_channels`. This is best done with the `CHANNEL_KEYWORD_OPTIONS` symbol in the `bitnet_domains_driver.com` procedure as described in Section 3.8. See Section 2.2 for details on and implications of the `nolocaluser` keyword.

3.11 Installing PMDF as the Local Mail Delivery (LMD) Agent

PMDF's slave program, `BN_SLAVE`, is designed to replace the `Jnet` or `ANJE` Local Mail Delivery agent. The PMDF replacement has the advantage that it generates addresses that can be replied to via PMDF (*i.e.*, all messages should be reply-able if PMDF is configured correctly) and messages can be routed through to other systems via PMDF automatically.

3.11.1 Jnet Local Mail Delivery

Normally the `Jnet` Local Mail Delivery agent is a process running a specially doctored version of `VMS MAIL`. In order to have PMDF deliver local mail, this process must run `BN_SLAVE` instead.

`BN_SLAVE` is established as the Local Mail Delivery agent by copying a new version of the file `lmd.com` to `JAN_SYS`. For `Jnet` versions 3.7, 3.6, and 3.5 use the command

```
$ COPY PMDF_COM:lmd.com JAN_SYS:lmd.com
```

For `Jnet` version 3.4, instead use the command

```
$ COPY PMDF_COM:lmd.v34 JAN_SYS:lmd.com
```

or, for `Jnet` version 3.3 or 3.2, use the command

```
$ COPY PMDF_COM:lmd.v33 JAN_SYS:lmd.com
```

If you are running `Jnet` on more than one node in a cluster, then be sure that the PMDF `lmd.com` (or `lmd.v34` or `lmd.v33`) file is seen by each `Jnet` node in the cluster (*i.e.*, ends up in the cluster common `JAN_SYS:` directory) or is placed in the specific `JAN_SYS:` directories of each individual `Jnet` node which is to run PMDF's local mail delivery daemon.

BITNET Channels (OpenVMS) Installing PMDF as the Local Mail Delivery (LMD) Agent

`lmd.com` can require alteration for use with other versions of Jnet. Be sure to compare PMDF's `lmd.com` with the one provided with Jnet and check for any discrepancies. The only real difference between the two should be that PMDF's `lmd.com` defines the logical names `PMDF_CHANNEL` and `SYS$OUTPUT` and then runs `BN_SLAVE` instead of defining the logical names `SYS$SCRATCH` and `JAN_MAILPROT` and then running `VMS MAIL`. The code in `lmd.com` to start up the `FANOUT` and `PROFS` daemons is not directly related to PMDF. The function of these daemons is described fully in the Jnet documentation.

`lmd.com` will also have to be edited if a name other than "bit_local" is used for the local Jnet channel. Change only the line that defines the `PMDF_CHANNEL` logical name.

Once this is done Jnet's local servers should be shut down and restarted to activate the new agent. The following commands will usually suffice, but see the *Jnet System Manager's Guide* for complete information on restarting Jnet.

```
$ RUN JAN_SYS:JCP
JCP> STOP local-host
JCP> START local-host
```

where `local-host` is Jnet's name for the system running Jnet and PMDF.

If the `BN_SLAVE` daemon should fail for any reason, the conventional Jnet Local Mail Delivery agent will be started instead. A mail message will also be sent to the `SYSTEM` account indicating that this has happened. The file `PMDF_LOG:bn_slave.log` should be examined to determine the cause of the problem if `BN_SLAVE` does fail.

For efficiency reasons the `BN_SLAVE` program only reads the PMDF configuration file once at startup. If the configuration file is edited or recompiled the `BN_SLAVE` program needs to be restarted. A restart can be requested with the OpenVMS command:

```
$ PMDF RESTART BN_SLAVE
```

`SYSLCK` privilege is required to issue this command which will restart all `BN_SLAVE` processes across your cluster. Restarting Jnet entirely is not necessary; this command alone is sufficient if only PMDF configuration information has changed. Note, however, that this command does *not* reload Jnet routing tables; consult the Jnet documentation to find out how to reload Jnet's tables.

Delivery via `BN_SLAVE` can be temporarily disabled by defining `PMDF_DISABLE_BN_SLAVE` as a system logical name (the translation value does not matter as long as the logical is defined). `lmd.com` will invoke the regular Jnet mail delivery mechanism if this logical is defined. To revert from PMDF delivery to regular Jnet delivery on a running system, first define this logical and then use the `PMDF RESTART BN_SLAVE` command to restart `BN_SLAVE`. The switchover will actually occur the next time a class M message is received by Jnet. In a cluster, you must define this logical on each cluster member to be affected. Should you later want to restart `BN_SLAVE`, deassign the `PMDF_DISABLE_BN_SLAVE` logical, and then stop and restart the local Jnet link with `JCP`, as shown above. (The `PMDF RESTART` command will not restart `BN_SLAVE` if there is no `BN_SLAVE` process currently running; it only restarts an active process.)

BITNET Channels (OpenVMS)

Installing PMDF as the Local Mail Delivery (LMD) Agent

To permanently shut down BN_SLAVE (until the system is rebooted or the Jnet link for the local node is restarted), you can use the PMDF SHUTDOWN command:

```
$ PMDF SHUTDOWN BN_SLAVE
```

This will cause all BN_SLAVE processes across a cluster to shut themselves down and exit. They will not “fail over” to the default Jnet LMD server: you are simply left without any LMD daemon running at all.

3.11.2 ANJE Local Mail Delivery

PMDF's ANJE slave program, BN_SLAVE, is designed to act as the ANJE delivery agent. BN_SLAVE is run from a symbiont executing in ANJE's receive queue when a file of the name PMDF_COM:bn_slave.com is found. The file contains:

```
$ set verify
$ ! bn_slave.com - Command file to run the slave component of the
$ ! BITNET channel ANJE_.
$ !
$ set noon
$ define/user pmdf_channel "anje_local"
$ define/user rqst_filespec 'pl'
$ run pmdf_exe:bn_slave.exe
$ status == ''$status''
$ set noverify
```

3.11.3 Envelope Creation in BN_SLAVE with Jnet

Jnet appends a special Received: line to all messages passed to BN_SLAVE. This Received: line contains both recipient and sender addresses for the message. No other envelope information is provided by Jnet; in particular, no envelope From: information is available; the Received: line only gives the sender of the message, which is often a mailer or list server of some kind and is not actually a From: address.

BN_SLAVE uses the recipient specified in the Received: line to construct the envelope To: address. However, the envelope From: address is constructed using a system similar to the one used to construct VMS MAIL From: addresses — the beginning of the message is parsed as an RFC 822 header and an attempt is made to use the fields from this header. The fields that are used, in order of decreasing priority, are:

1. Resent-Sender: (if present)
2. Sender: (if present)
3. Resent-From: (if present)
4. From: (if present)
5. Resent-Reply-To: (if present)
6. Reply-To: (if present)

BITNET Channels (OpenVMS) Installing PMDF as the Local Mail Delivery (LMD) Agent

7. Address in the Received: line (always present)

Note that the envelope is discarded and this processing is irrelevant if the message is actually a BSMTP command stream in route to PMDF's mailer implemented in the BITNET gateway channel (see the description of the BITNET gateway channel below).

3.11.4 Handling Usernames Longer Than Eight Characters

This section is only of interest to sites *not* running a BITNET “mailer” (*i.e.*, BITNET gateway or trusted mailer). Note that the January 1, 1992 CREN regulations require BITNET sites wanting to send mail to the Internet to run a mailer. Process Software recommends that all sites register themselves as a mailer and configure their PMDF to operate as a BITNET gateway.

NJE can only deal with mailbox names containing 8 or fewer characters. OpenVMS, on the other hand, allows usernames with up to 12 characters in them (and supports mail forwarding aliases up to 31 characters long).

Jnet deals with this problem by delivering mail to the first user with a name that matches in the first 8 characters — even if there is more than one username which matches in the first 8 characters. Note that this scheme, while effective in some cases, is not a general solution to the problem. In particular, it does not handle mailboxes that do *not* correspond to an actual user on the system (*i.e.*, an alias) properly.

PMDF's local mail delivery facility also does do this type of matching, but only as a last resort. When PMDF encounters a potentially truncated mailbox name it scans appropriate lines in the message header (first the X-Envelope-to:, in the hopes that the message originated with another PMDF mailer, and then the Resent-to:, Resent-cc:, To:, and Cc: lines) looking for an address that could have been the one that was truncated. The comparison is done very cautiously in order to limit the possibility of a spurious match. If a matching address is found it is used in lieu of the truncated address. If this process fails the Jnet strategy is used instead with one notable and important exception: if the truncated username matches the first 8 characters of more than one username in the SYSUAF then the mail is bounced as undeliverable rather than run the risk of delivering to the wrong user.

This mechanism is still not entirely satisfactory. There is no guarantee that the address of the addressee is actually present in the message header! So PMDF offers an additional technique for dealing with long usernames: Set up a local mailer (BITNET gateway channel) and use it for incoming mail traffic. Gateways do not inherit the 8 character name length restrictions of NJE. This is the preferred solution in all cases.

One final possibility is to simply require all usernames and aliases to be 8 characters or less in length. This is usually not practical unless it can be done at the time the system is initially configured. In this case it is recommended for systems that are going to make extensive use of Jnet.

BITNET Channels (OpenVMS)

Routing to Systems Not Running NJE

3.12 Routing to Systems Not Running NJE

An NJE network provides a flat, closed namespace within which every system on the network must know how to reach every other. This presents problems when systems that are not running NJE must be accommodated. When large external networks are involved true gateway facilities must be used to reach them. Using PMDF to provide and access such facilities is described in subsequent sections of this document.

However, if only a few non-NJE systems are to be made accessible to the NJE world, it is possible to actually enter these systems into the NJE namespace even though they are not running an implementation of NJE. PMDF then acts as the routing element that takes messages from NJE and routes them to the non-NJE systems. The non-NJE systems can be any type of system with which PMDF is capable of communicating.

This trick depends on modifying Jnet's internal router and only provides mail access (sending files and interactive messages is not possible), so it is not a complete substitute for actually connecting systems directly to NJE. This technique is not applicable to ANJE.

The following steps must be taken to provide a route from the NJE network to a non-NJE system:

1. The non-NJE system must be established in the routing tables of ALL the systems on the NJE network. Messages to the non-NJE system should be routed to the system running both Jnet and PMDF that is going to act as the bridge between NJE and whatever network the non-NJE system uses.
2. A modified version of `mfsdisp.exe` (the message routing component of Jnet) must be installed on the bridge system. `mfsdisp.exe` is invoked when a system name is encountered that does not correspond to a known route or link. This module is intended to be customizable, so FORTRAN source for `mfsdisp.exe` is provided in the file `JAN_ROOT:[lib]mfsdisp.for`. A command file to rebuild `mfsdisp.exe` is provided as `JAN_ROOT:[lib]buildmfsdisp.com`.

The portion of `mfsdisp.for` that must be modified is the legality check for the destination system name. In Jnet 3.2 the code for this check is:

```
! Check to see if it was for this site or this cluster
ELSEIF ((HOST .NE. TO_NODE) .AND. (CLUSTER .NE. TO_NODE)) THEN
  RECEIVE = .TRUE.
  RECEIVE_USER = DEAD_USER
  NOTIFY = .FALSE.
```

This essentially says that if the destination system `TO_NODE` is unknown the message is sent to `DEAD_USER` (usually the local postmaster). This check must be modified to accommodate the non-NJE system as a legal local system. For example, suppose the name of the non-NJE system is placed in the `CHARACTER*8` variable `SPECIAL_NODE`. Then the check could be modified to read:

```
! Check to see if it was for this site or this cluster
ELSEIF ((HOST .NE. TO_NODE) .AND.
1      (CLUSTER .NE. TO_NODE) .AND.
2      (SPECIAL_NODE .NE. TO_NODE)) THEN
  RECEIVE = .TRUE.
  RECEIVE_USER = DEAD_USER
  NOTIFY = .FALSE.
```

Note that the continuation lines must begin with tabs and not with spaces. Be sure

BITNET Channels (OpenVMS) Routing to Systems Not Running NJE

to incorporate a value for SPECIAL_NODE into mfsdisp.for by defining it as a character variable:

```
CHARACTER*8 SPECIAL_NODE /'NONJE'/
```

Here NONJE is, of course, the name the non-NJE system is known by in the NJE world.

An additional change to mfsdisp.for can be necessary if the system being added supports usernames longer than 8 characters. Jnet will expand usernames in the message envelope using the local host's system authorization file unless you tell it not to. This can be avoided by adding code such as the following to mfsdisp.for. This code should be inserted after the point where mfsdisp.for has decided to handle the file as a mail message:

```
! Try to get just the first 8 char user_id on
! the received line so that PMDF will use the
! original ID as the envelope passed to the
! remote system for the pseudo BITNET host hack.
IF ((SPECIAL_NODE .EQ. TO_NODE) THEN
    USERNAME = TO_USER
END_IF
```

This code resets the expanded username to be used in the Received: line back to the original userid from the file tag.

Once the modifications are made, mfsdisp.exe must be rebuilt by running buildmfsdisp.com. After mfsdisp.exe has been rebuilt, it must be activated by shutting down and restarting the local Jnet server. (This is described in the section above on installing PMDF's local mail delivery agent.)

Note that this example only shows a very simplistic way to add a single system to mfsdisp. This technique is only suitable for adding a limited number of systems. If a lot of systems are to be attached in this manner mfsdisp should be modified to consult some sort of database of systems reachable via PMDF.

3. PMDF's local mail delivery agent will now be activated to process messages sent to the non-NJE system. The name used by NJE for this system must be defined in PMDF's configuration file. Note that since dotted domain names are not supported by NJE, the name must be a simple dotless shortform name. Of course, PMDF's configuration file can map the shortform name into a full domain name if desired.
4. At this point it should be possible to send messages from the NJE network to the non-NJE system. In order for these messages to be reliable the non-NJE system must be configured to forward all messages headed to the NJE network to the bridge system. The manner in which this is done will depend on the type of non-NJE system involved.

It should be emphasized once again that this trick of placing non-NJE systems within the NJE namespace does not provide full NJE functionality, nor does it provide the expandability that a true gateway should provide. However, this technique can prove useful when only small heterogeneous networks are involved.

BITNET Channels (OpenVMS)

Local NJE Mailers and Mailer Gateways

3.13 Local NJE Mailers and Mailer Gateways

PMDF also contains facilities to implement a mailer or mailer gateway of its own. A mailer is typically used to provide support for longer usernames than BITNET normally supports. Mailer gateways can also be used to provide access to local non-BITNET systems from remote BITNET sites. Systems not directly attached to BITNET cannot be entered in the BITNET routing tables and therefore cannot have a BITNET address. The only way to provide remote BITNET access to such systems is to implement a local mailer which uses information in the body of the message to route the message to its true destination.

The mailer provided by the PMDF BITNET gateway channel uses BSMTP (batch SMTP) exclusively. A mailer is really nothing more than a BSMTP processor, with the proper facilities for extracting BITNET BSMTP material from an enclosing message wrapper.

PMDF's local mailer facility is implemented as a separate channel program called BN_GATEWAY.

Note: Setting up a local mailer is a good idea. However, it is pointless to set up a mailer without registering it, since nobody will know that it is there if it isn't registered, so registering it is also a good idea.

3.13.1 Local Mailer Operation

The operation of the mailer is quite simple. An alias is placed in the PMDF alias file that forwards messages sent to a special mailbox name (usually MAILER) to the BITNET gateway channel. BN_GATEWAY then runs as the channel master program. It discards the PMDF envelope and the initial RFC 822 header on the message. The message body is passed to the SMTP command interpreter for execution. The message is deleted after it has been executed.

3.13.2 Setting Up a Local Mailer

Note: If you generated your configuration with the PMDF CONFIGURE utility and answered "yes" to the question, "Are you a BITNET gateway", then a local mailer was configured for you and you need not undertake the steps discussed in this section. The usage of the PMDF configuration utility is described in the OpenVMS edition of the *PMDF Installation Guide*.

The first step in creating a local mailer is to create a channel table entry for the BN_GATEWAY program. The proper format for this entry is:

BITNET Channels (OpenVMS) Local NJE Mailers and Mailer Gateways

```
bit_gateway  
bitnet-gateway
```

Your mailer can not function properly if this format is not followed exactly.

The next thing to do is to place an entry for the mailer mailbox name in the PMDF alias file. Edit the file `PMDF_ALIAS_FILE` (on OpenVMS this logical name usually translates to `PMDF_TABLE:aliases.`) and add the following line:

```
mailer: mailer@bitnet-gateway
```

The use of the mailbox name `MAILER` is recommended (since it is in common use on BITNET) but any name can be used.

It is also a good idea to have a rewrite rule that maps the mailer name to the proper channel. This is only required if the match-all rule, “.” is used, but it will not hurt to have it in any case:

```
bitnet-gateway      $U@bitnet-gateway
```

Once these changes have been made any mail sent to `MAILER` on the local host will be interpreted as a series of SMTP commands. In other words, your local mailer is now operational. You can test your mailer manually by sending a message containing a series of SMTP commands to `IN%"MAILER"` using `VMS MAIL`. For example, the following set of commands sends a test message to the local postmaster:

```
HELO test.system  
MAIL FROM:<postmaster>  
RCPT TO:<postmaster>  
DATA  
From: postmaster  
To: postmaster  
Subject: Testing the local mailer  
This is a test message to exercise the local mailer.  
.  
QUIT
```

The last thing to do is to register your mailer so that remote sites will know about it. The following section explains how this is done.

3.13.3 Registering a Mailer or Mailer Gateway on BITNET

If you choose to configure a BITNET gateway channel and thus have a mailer or mailer gateway on your system, this fact must be registered in a `:servers1` tag in your BITNET node entry or entries. (This tag is what is used to generate the contents of the `:mailer` entries in the `XMAILER NAMES` file.)

In general the `:servers1` tag describes the network services provided by server software to the node in which the tag appears. Currently the only valid server that can be defined is a `MAIL` server, or mailer. The format of the data associated with this tag is:

BITNET Channels (OpenVMS)

Local NJE Mailers and Mailer Gateways

```
:servers1.serverid(servertype,formats,classes,attributes,contacts)
```

An example of a typical `:servers1` tag is:

```
:servers1.MAILER@YMIR(MAIL,PU,M,BSMTP,P_POSTMAST)
```

The *serverid* is the network address of the mailer in the form *USERID@NODE*. Messages to be processed by the mailer are sent to this address. You must specify both the *USERID* and the *NODE*. The *USERID* should match whatever name you assigned to your mailer in the alias file. *MAILER* is the recommended userid for both PMDF and the VM Mailer, but *SMTPUSER* is also a common choice.

The *servertype* is the type of server. Currently the only valid type is *MAIL*, meaning that the server is a mailer.

The *formats* entry lists the NJE file formats accepted by this server. The acceptable formats depend on the underlying network transport and not PMDF. Both *Jnet* and *ANJE* support both *PUNCH* files and *NETDATA* format, but most mailers will only send and accept messages in *PUNCH* format. To indicate *PUNCH* format, the default format, specify *PU*. Other formats are described in the *NEWTAGS DESCRIPT* file available from *LISTSERV@BITNIC*.

The *classes* field lists the NJE file classes accepted by this server. The valid classes are the letters A-Z or numbers 0-9. Class *M* files are customarily recognized as mail and class *M* is the default.

The *attributes* field lists message encoding that the mailer accepts. For PMDF this field should always be *BSMTP*. *BSMTP* is the default.

The *contacts* field defines the person responsible for the mailer and who is to be contacted in case any problems arise. The tag used in this entry must be defined in the node entry or in a higher level site or member entry. *This entry is not an address in its own right! It is a pointer to a tag defined by another part of the node entry!*

If you would like to accept all the defaults simply specify:

```
:servers1.MAILER@NODE(MAIL,,,,)
```

where *NODE* is the BITNET node name for your node.

If you have a mailer or mailer gateway on one node which serves other nodes then you can add a `:servers1` tag to the other node entries and reference the mailer at the first node. For example, if you have nodes *ABC* and *XYZ* and have a mailer on *ABC* which also serves node *XYZ*, you would add a `:servers1` tag to node *XYZ* and specify *MAILER@ABC* as the *serverid*.

The *NEWTAGS DESCRIPT* file, available from *LISTSERV@BITNIC* describes the `:servers1` tag in more detail, as well as all other tags in *BITEARN NODES*.

Once you have decided how to define the `:servers1` tag for your node entry or entries you must submit an update request to *UPDATE@BITNIC*. The following is an example of an update job to add or modify the `:servers1` tag for the node *XYZ*; (this information is not case sensitive):

BITNET Channels (OpenVMS) Local NJE Mailers and Mailer Gateways

```
modify node xyz  
:servers1.MAILER@XYZ(MAIL,PU,M,BSMTP,P_POSTMAST)
```

or

```
modify node xyz  
:servers1.MAILER@XYZ(MAIL,,,,)
```

You can submit this job by using electronic mail or as a file. For more information about updating your node entry using UPDATE@BITNIC, refer to the document *Update Procedure*, available from LISTSERV@BITNIC.

If you have any questions about or problems with updating your node entry contact the BITNET administrators on BITNIC for additional assistance.

3.13.4 Registering Your NJE Gateway to Serve an Entire Domain

If your BITNET gateway provides service to an entire domain, that domain must be registered with BITNIC so other BITNET mailers will know how to reach it. The procedure for doing this is outlined in the *Domain Guide* document, available from LISTSERV@BITNIC:

```
$ SEND LISTSERV@BITNIC SEND DOMAIN GUIDE
```

4 PMDF-XGS

Note: PMDF-XGS is no longer supported as of PMDF version 6.2. This chapter was originally part of the *PMDF System Manager's Guide*.

PMDF-XGS provides an e-mail connection between systems using the SNADS (SNA Distribution Services) protocol and the myriad of networks and mail systems reachable with PMDF. Mail systems that use SNADS include:

- IBM OV/MVS with DISOSS
- IBM OV/400
- Verimation's Memo

running on IBM mainframes and mid-range machines.

PMDF-XGS consists of two major components. The first component consists of PMDF-XGS channel programs running on a PMDF system — an OpenVMS, Tru64 UNIX, or Solaris system. The second component is the PMDF-XGS transport bridge, running on an OS/2 or NT (Intel) system. The PMDF-XGS transport bridge on the OS/2 or NT system acts as a pipeline, with a SNADS connection to the SNADS nodes in one direction, and a TCP/IP connection to PMDF-XGS on the PMDF system in the other direction.

The PMDF-XGS transport bridge (the OS/2 or NT system) appears to the SNADS network as an AS/400 SNADS node, and the PMDF/PMDF-XGS system itself appears to the SNADS network as another SNADS node reachable through the PMDF-XGS transport bridge. Other host and mail systems reachable through PMDF, *e.g.*, other SMTP hosts or PC mail systems, appear to the SNADS network as other SNADS nodes which are reached through the the PMDF-XGS transport bridge in its guise of SNADS node. And from the PMDF side, each SNADS node appears as an additional host, akin to an additional SMTP host. Section 4.5 below shows an example site.

Note: SNADS channels are provided by the PMDF-XGS product, not PMDF proper. PMDF-XGS is a layered product built on top of PMDF. It is licensed separately from the base PMDF product.

4.1 Required Hardware and Software Versions

The PMDF/PMDF-XGS system must have a TCP/IP network stack.

The PMDF-XGS transport bridge component currently runs on either an OS/2 system or an NT (Intel) system.

PMDF-XGS

Required Hardware and Software Versions

If running the PMDF-XGS transport bridge component on an OS/2 system, that OS/2 system requires a TCP/IP stack and the IBM Communications Manager/2 for the SNA link. That OS/2 system must have one of:

- OS/2 V2.1 plus IBM TCP/IP (base), or
- OS/2 Warp plus IBM TCP/IP (base), or
- OS/2 Warp Connect,

and must also have:

- IBM Communications Manager/2 V1.11, part number 33H7312, or
- IBM eNetwork Personal Communications V4.2, part number 4074556.

If running the PMDF-XGS transport bridge component on an NT (Intel) system, that NT system requires:

- IBM Personal Communications V4.1 or later, part number 4074548, or
- IBM eNetwork Personal Communications V4.2, part number 4074556.

4.2 Background SNADS Concepts

This section provides an overview of SNADS e-mail concepts and how they relate to e-mail concepts in the PMDF world. While readers familiar with the SNADS environment can prefer to skip this section, the information presented here can provide a useful context for readers unfamiliar with SNADS.

4.2.1 SNADS Architecture

SNADS is a very different mail protocol from SMTP. A SNADS mail item is called a *distribution*. There are two types of distribution: data distributions and status distributions.

A *data distribution* consists of three parts, roughly corresponding to envelope information, headers, and message body. More precisely, these parts are:

1. A *SNADS command*, which is envelope information: originator, recipients and message id;
2. A *DIA profile* (Document Interchange Architecture profile), which is structured header information; and
3. A *DIA document*, which is either FFT (Final Form Text — simple EBCDIC text), RFT (Revisable Form Text — a word processing format), or a binary file.

A *status distribution* is expected by SNADS user agents for each recipient of each data distribution. These are very structured messages, and there is no equivalent in SMTP until the new NOTARY recommendations are widely implemented.

PMDF-XGS Background SNADS Concepts

A SNADS user has a name consisting of two parts, each of which can be no more than eight characters long. These two parts are called the DEN (Distribution Element Name), and DGN (Distribution Group Name). A SNADS user name is also referred to as a DUN (Destination User Name).

Each SNADS node also has a name which has one or two parts. The first part is called the REN (Routing Element Name). The second part is called the RGN (Routing Group Name). The RGN can be absent, and often is. The SNADS node name is also referred to as a DSUN (Distribution Service Unit Name).

SNADS user names, *i.e.*, DUN's, must be unique throughout the SNADS network. A DUN is unique and thus provides, in effect, a complete address, even without the DSUN. That is, a SNADS user can be addressed using merely the user DEN and DGN information; no node information, *i.e.*, no REN or RGN information, is required. The theory is that user names should not have to change if the users move from being served by one machine to another. In practice, it is almost universal practice to constrain user names so that the DGN of each user is the same as the REN of the machine serving them, and to not use the RGN. Effectively, this yields an eight character by eight character addressing space consisting of merely the DEN and REN, or *username@nodename*. Thus, for example, a SNADS user FRED@MVS21 will probably be found on the machine called MVS21.

The SNADS command in a data distribution (*i.e.*, message) contains the DEN, DGN, REN and RGN for each recipient. SNADS works by routing the distribution using the REN and RGN information only to reach the destination node, and then using the DEN and DGN to pass the distribution to the user. If a SNADS distribution is routed (using the REN and RGN) to the wrong node, then that node will redirect (using the DEN and DGN) to the right DSUN. In theory, this means that each node has to be configured to know where each user is. In practice, (with DGN and REN constrained to be the same and RGN not used), directory information of the form "users with DGN=XXX are found at REN=XXX" is sufficient.

4.3 PMDF-XGS Architecture

This section describes the architecture of PMDF-XGS; in particular, the structure of PMDF-XGS vis-à-vis SNADS.

4.3.1 SNADS Addresses in PMDF-XGS

PMDF-XGS maps the DEN part of a DUN into an SMTP user name, and the DGN part of a DUN into a short-form host name, *e.g.*, a short-form SMTP host name. Each PMDF-XGS SNADS channel, *e.g.*, the *snads_local* channel, has associated with it the name of an immediately adjacent DSUN or SNADS node, and the necessary information on how to reach that node. (Once a message from PMDF to the SNADS world makes it to that immediately adjacent SNADS node, then SNADS routing takes over for getting the message to the intended final SNADS destination.)

PMDF-XGS

PMDF-XGS Architecture

PMDF has a number of facilities that can be used to assist in the transformation between RFC 822 style addresses and SNADS address; see Section 4.4 for details.

4.3.2 The Topology of the PMDF-XGS Connection to the SNADS World

In a simple PMDF-XGS gateway configuration, there is one SNADS channel, `snads_local`, which connects via the PMDF-XGS transport bridge to one immediately adjacent SNADS node; this effectively is a connection to the entire SNADS world for which that adjacent SNADS node can route messages. Distributions (messages) from the SNADS world addressed to PMDF are routed within the SNADS world first to the SNADS node adjacent to the PMDF-XGS transport bridge and then out through the PMDF-XGS transport bridge to the PMDF-XGS/PMDF system via the `snads_local` channel. (Note that although messages from PMDF to the SNADS world can physically be being routed through a particular SNADS node, that is purely a SNADS network issue and is invisible from the PMDF side. This is similar to the way that all messages from SNADS pass through the PMDF system, even if the message is destined for some other SMTP host or other mail system to which PMDF provides access.)

In a more complex PMDF-XGS gateway configuration, there can be many SNADS channels, each associated with a different SNADS node (DSUN). In this case, each message (distribution) going to the SNADS world will be sent directly to the DSUN serving the message recipient over the respective `snads_ren` channel (where *ren* is the REN portion of the DSUN), and distributions (messages) coming from the SNADS world to PMDF will be transmitted back through that `snads_ren` channel. The use of such a configuration is mainly an issue of efficiency and convenience at the SNADS network level; to cut down on SNADS network traffic, or for administrative reasons, it can be desirable to have SNADS nodes communicate directly with the PMDF-XGS transport bridge rather than routing their distributions to the PMDF-XGS transport bridge through another SNADS node.

4.3.3 The Function of the Transport Bridge

Note that the PMDF-XGS transport bridge performs no SNADS routing. This is true even in the case (multiple SNADS channels) where the PMDF-XGS transport bridge is directly adjacent to several SNADS nodes; when multiple channels are used, the PMDF system itself handles routing messages to appropriate channels. The PMDF-XGS transport bridge is purely a pipeline converter, taking SNADS distributions and squirting them over TCP/IP to PMDF, or receiving messages over TCP/IP from PMDF and converting them into SNADS distributions which it squirts to the SNADS world. The PMDF-XGS transport bridge, although it is configured to appear from the SNADS side as a SNADS node itself, and to accept special TCP/IP connections from the PMDF side, is effectively transparent from the e-mail point of view. The PMDF-XGS transport bridge never holds any mail: it has no internal queues.

At the IP level, the PMDF-XGS transport bridge system needs its own name . And for clarity, the discussion in this documentation gives the PMDF-XGS transport bridge system its own SNADS node name distinct from the apparent SNADS node name used for the PMDF-XGS/PMDF system itself. However, in point of fact the PMDF-XGS transport bridge system and the actual PMDF-XGS/PMDF system itself are effectively the same “SNADS node” from the SNADS point of view, and you can, if you want, use one SNADS name for both.

When a mail item is given to the PMDF-XGS transport bridge by PMDF, the PMDF-XGS transport bridge converts it, passes it on to the SNADS node associated with the channel, and waits for the SNADS node to accept responsibility for the mail before telling PMDF that the transfer is complete. When a SNADS distribution is received from SNADS, the distribution is converted to SMTP and passed immediately to PMDF: only when PMDF has accepted responsibility for the mail will the PMDF-XGS transport bridge confirm to the sending SNADS system that the transfer has been completed. That is, the PMDF-XGS transport bridge is not a destination point for messages and does not accept messages on its own behalf or perform routing of messages; it merely passes messages straight through in either direction. Hand-off of responsibility for the messages occurs between the PMDF system and the SNADS nodes; the PMDF-XGS transport bridge acts as an intermediary in relaying the confirmation that a message has been received from one side to the other, but does not accept the message itself, and thus at any instant either the PMDF system or a SNADS node has primary responsibility for a message (and a copy of the message).

4.3.4 Message Attachments

MIME messages can have messages of complex and arbitrary structure. The SNADS world, on the other hand, does not allow for messages with attachments. So PMDF has to perform a message structure conversion when sending to SNADS. Messages with multiple parts are “flattened”, with text markers inserted to show the original message structure, and any binary attachments are split out and sent as individual separate messages.

4.3.5 Notification/status Messages

When sending to the SNADS world, PMDF-XGS converts SMTP notification requests (requests for read or delivery receipts) to SNADS status requests; then when the SNADS side sends the status distribution back, PMDF-XGS converts it to an SMTP notification message.

The SNADS side expects a status distribution back for each recipient of a data distribution (message). Since the return of such receipt messages can not in general be guaranteed from the SMTP/RFC 822 side — the generation of responses to such requests is entirely up to the eventual receiving mailer and user agent, and while some mailers and user agents support them, others do not – the PMDF-XGS gateway sends a status distribution (message) back to the SNADS sender when the gateway first receives the message destined for the SMTP/RFC 822 side. That is, SNADS senders will receive a

PMDF-XGS

PMDF-XGS Architecture

status distribution from the PMDF-XGS gateway itself not, in general, from the eventual message recipient.

4.4 Addressing Solutions

As the SNADS world is effectively limited to eight character by eight character addressing, while the RFC 822 world has arbitrarily long usernames and domain names, the issue of address transformations must be addressed, particularly how to represent RFC 822 addresses within the SNADS world. There are four general approaches which can be used with PMDF-XGS singly or in combination.

- PMDF can be (and generally is) configured to convert specified short form host names into fully qualified domain names. Thus if an RFC 822 address uses no more than eight characters in the user name and no more than eight characters in the host name part of the domain name, then a natural setup is to simply represent the RFC 822 address on the SNADS side in truncated form, with merely the user name and host name.
- PMDF can be configured to use pseudodomain aliases for specified RFC 822 domain names, *e.g.*, “comp” for “complicated.subdivision.com”. Or if long usernames are an issue, PMDF can be configured to look up short aliases in a database or databases to be transformed into actual long addresses.
- PMDF can be configured to “autoregister” addresses going to SNADS. That is, when PMDF sees a “long” address going to SNADS, PMDF can automatically generate a unique short form address, substituting the generated short form in as the From: address on the message going to SNADS, and looking that short form up and substituting the true “long” address back in if a SNADS recipient replies using that short form address. For more information, see Chapter 3 in the *PMDF System Manager’s Guide*.
- PMDF’s addressing channel can be used as a way to let SNADS users send to any arbitrary address reachable through PMDF. SNADS users can send a message to, *e.g.*, MAILMAN@PMDFnode, with the actual long form RFC 822 recipient address embedded in the text of the message and specially marked for PMDF’s use, and PMDF will pull the real address out of the message text and resend the message to that specified address. See Section 27.1 in the *PMDF System Manager’s Guide* for details on the format for embedding such addressing information in the text of the message.

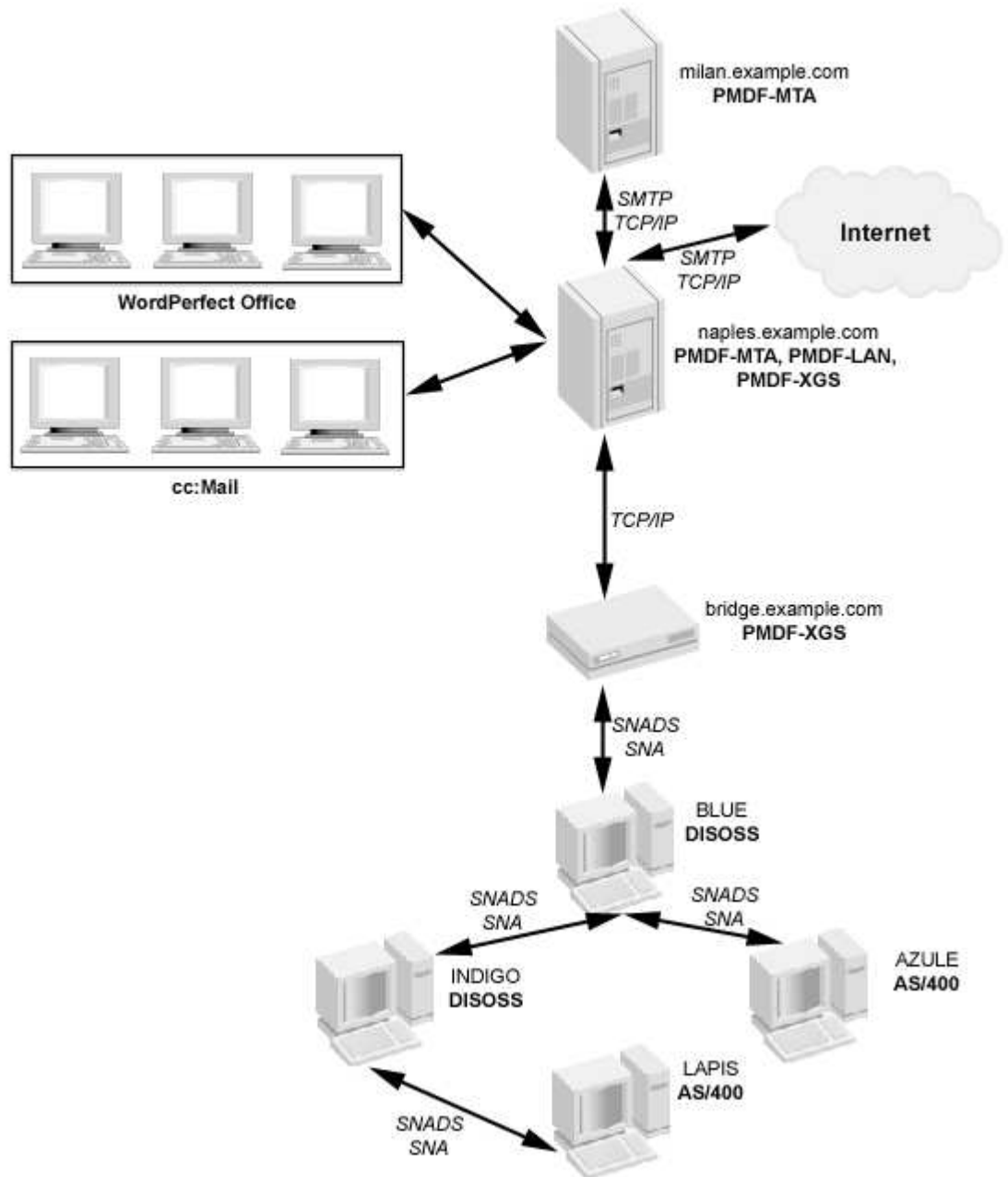
4.5 Example Site Using PMDF-XGS

This is best shown by example. Consider the network shown in Figure 4–1. A snads_local channel would connect the PMDF-XGS system `naples.example.com` with the SNADS world, handing off messages to the SNADS node BLUE, by way of the PMDF-XGS transport bridge `bridge.example.com`.

PMDF-XGS

Example Site Using PMDF-XGS

Figure 4-1 Sample SNADS and SMTP Site



On the PMDF side, one would add MX records for pseudodomain names such as `blue.example.com`, `indigo.example.com`, `azule.example.com`, and `lapis.example.com` pointing to the PMDF-XGS system `naples.example.com`.

PMDF-XGS

Example Site Using PMDF-XGS

On the SNADS side, one would at a minimum configure the SNADS nodes BLUE, INDIGO, AZULE, and LAPIS to believe that the PMDF-XGS transport bridge `bridge.example.com` is another SNADS node, and that `naples.example.com` is another SNADS node reachable through `bridge.example.com`. One would probably also configure the SNADS nodes to believe that `milan` is an additional SNADS nodes reachable through `bridge.example.com`.

That is, on the SMTP side, the SNADS nodes BLUE, INDIGO, AZULE, and LAPIS would appear to be additional SMTP systems `blue.example.com`, `indigo.example.com`, `azule.example.com`, and `lapis.example.com` reachable through `naples.example.com`; on the SNADS side, the SMTP systems `naples.example.com` and `milan.example.com` would appear to be additional SNADS nodes. NAPLES and MILAN would be reachable through the NAPLES transport bridge in its guise of SNADS node.

Then using the standard naming conventions, user `carol` at `milan.example.com` can address a user CHARLES at INDIGO as `charles@indigo.example.com` (or perhaps even as `charles@indigo`, if the TCP/IP package and PMDF system are configured to allow use of short form names). User BOB at AZULE can address mail to `betty` at `naples.example.com` as `BETTY@NAPLES`, but will not be able to address `christopher` at `milan` directly, because SNADS names are limited to eight characters. This is solved by sending the mail to the PMDF addressing channel by addressing the mail to `MAILMAN@NAPLES`, and embedding the addresses of the real recipients in the text of the note. This also applies to recipients at SMTP hosts for which PMDF has not been configured to convert the short form host name into a fully qualified domain name, or whose host name is over eight characters long. For instance a SNADS user could not directly address `priscilla` at `prometheus`, or `susan@styx.hades.org`.

4.6 Configuring PMDF-XGS

This section describes configuring PMDF-XGS. There are three major parts to configure: (1) configuring each SNADS node to know about the PMDF-XGS transport bridge as another SNADS node, (2) configuring PMDF-XGS on the transport bridge, (3) configuring PMDF-XGS on the PMDF system.

Each SNADS node must be configured to believe that the PMDF-XGS transport bridge is another SNADS node and that the actual PMDF system is a SNADS node reachable through the PMDF-XGS transport bridge. And each SNADS node must be configured to be able to route distributions to the PMDF-XGS transport bridge, either directly or indirectly. Section 4.6.1 describes the configuration process on a DISOSS node, and Section 4.6.2 describes the configuration process on an AS/400 node. Note that once the PMDF-XGS transport bridge and the PMDF system are addressable from SNADS, then any host or mail system reachable by PMDF can be addressed from SNADS via aliased or embedded addressing; however, it is usually preferable to add additional routing information while configuring the SNADS nodes to allow direct addressing from SNADS of other commonly addressed hosts or mail systems. *The PMDF-XGS transport bridge looks to other SNADS nodes just like an AS/400 running OV/400. Other systems reachable through PMDF appear to the SNADS nodes as remote AS/400 systems.*

The PMDF-XGS transport bridge must be configured with a SNADS link to the SNADS world and a TCP/IP to the PMDF system. Section 4.6.3 describes this procedure.

PMDF-XGS Configuring PMDF-XGS

And PMDF-XGS must be configured on the PMDF system. Section 4.6.4 describes this procedure.

Note that the subsections below focus on the case of using one `snads_local` channel to connect PMDF and the SNADS world. In this case, all SNADS distributions, *i.e.*, all messages addressed to any SNADS recipient, will be sent from the PMDF system through the PMDF-XGS transport bridge to a single SNADS node, the default SNADS node associated with the channel, and that SNADS node will then have responsibility to route the distributions on to other SNADS nodes. Similarly, each message addressed to PMDF from the SNADS world will be routed within the SNADS world to the SNADS node which can pass the message through the PMDF-XGS transport bridge to the PMDF system. For example, in Figure 4–1, all SNADS nodes route their mail to the SMTP network through the node `BLUE`. However, it is also possible to use multiple SNADS channels connecting, through the PMDF-XGS transport bridge, directly to different SNADS nodes, which can be a more efficient setup for certain SNADS network topologies; see Section 4.8 for details.

4.6.1 Configuring a DISOSS Node

Each SNADS node must be configured to know about the PMDF-XGS transport bridge and about every host or mail system on the PMDF side which is to be directly addressable from the SNADS side. That is, at a minimum, each SNADS node must be configured to believe that the PMDF-XGS transport bridge is another SNADS node and that the PMDF system is a SNADS node reachable through the PMDF-XGS transport bridge. Once the PMDF system is reachable from SNADS, then any host or mail system reachable by PMDF can be made reachable from SNADS via aliased or embedded addressing; however, it is usually preferable to add additional routing information to the SNADS nodes to allow direct addressing from SNADS of other commonly addressed hosts or mail systems. *The PMDF-XGS transport bridge looks to other SNADS nodes just like an AS/400 running OV/400. Other systems reachable through PMDF appear to the SNADS nodes as remote AS/400 systems.*

The simplest way set up the SNADS nodes is to define directory information on each of the SNADS nodes so that users `*ANY` at `PMDFnodeDGN`, and users `*ANY` at `othersystemreachedthroughPMDF`, are all at the PMDF-XGS transport bridge (which appears to be another SNADS node), and then to define to the SNADS nodes how to send mail to the PMDF-XGS transport bridge system. `PMDFnodeDGN` is technically the PMDF node's DGN, from the SNADS point of view; however, as we consider the PMDF node's REN to be the same as its DGN, it is the same value as the PMDF node's REN (and thus effectively is the PMDF node name from the SNADS point of view).

There are three parts to configuring a DISOSS node to connect to PMDF-XGS. On each DISOSS node, you must add directory information specifying that the user location for the PMDF system (and any other system reachable through PMDF that you want to add) is on the PMDF-XGS transport bridge; see Section 4.6.1.1. You must inform each DISOSS node how to route distributions to the PMDF-XGS transport bridge; see Section 4.6.1.2. And you must configure the SNA connection, including both the VTAM and CICS definitions; see Section 4.6.1.3.

PMDF-XGS

Configuring PMDF-XGS

4.6.1.1 Adding Directory Information on a DISOSS Node

For a DISOSS node, the user location information is configured using the commands in Figure 4–2.

Figure 4–2 Defining SMTP Hosts to DISOSS

```
ADD    USERTYPE='REMOTE'
        DDN='PMDFnodeDGN'
        SA='*'
        RGN='          '
        REN='bridgeREN'.
ADD    USERTYPE='REMOTE'
        DDN='anothersystemreachedthroughPMDF'
        SA='*'
        RGN='          '
        REN='bridgeREN'.
ADD    USERTYPE='REMOTE'
        DDN='yetanothersystemreachedthroughPMDF'
        SA='*'
        RGN='          '
        REN='bridgeREN'.
...

```

where *PMDFnodeDGN* is the name you will use on the SNADS side for the PMDF system, and where *bridgeREN* is the name you will use on the SNADS side for the PMDF-XGS transport bridge.

4.6.1.2 Adding Routing Information on a DISOSS Node

Each DISOSS node needs to be told how to route messages to the PMDF-XGS transport bridge. Exactly where a particular DISOSS node should route a distribution eventually intended for the PMDF-XGS transport bridge will depend upon the topology of your SNADS network. The SNADS node directly adjacent to the PMDF-XGS transport bridge will need a new entry in its routing table coordinating the REN (or SNADS node name) of the PMDF-XGS transport bridge with the SNADS node's CICS definition of the PMDF-XGS transport bridge LU (Logical Unit). A SNADS node which is not adjacent to the PMDF-XGS transport bridge, but which presumably does already have an entry specifying how to get to the SNADS node which is adjacent to the PMDF-XGS transport bridge, will also need a new entry in its routing table for the PMDF-XGS transport bridge similar to its current entry for the SNADS node adjacent to the PMDF-XGS transport bridge.

If the DISOSS node is directly adjacent to the PMDF-XGS transport bridge, then it will need a new routing table entry along the lines of Figure 4–3. Here *bridgeREN* is the name of the PMDF-XGS transport bridge, and *bridgeCICSname* is the CICS name for the LU for the PMDF-XGS transport bridge, matching the name to be used during the CICS configuration step as described in Section 4.6.1.3.2 below; see particularly Figure 4–7 and Figure 4–8.

Figure 4–3 Defining the Route to the PMDF-XGS Transport Bridge on an Adjacent DISOSS Node

```
ADD    RGN='          '  
       REN='bridgeREN'  
       TRANSID='DSVE'  
       SSL='*'  
       QUEUE=bridgeCICsname
```

If the DISOSS node is not adjacent to the PMDF-XGS transport bridge, then it will need a new routing table entry on how to get to the PMDF-XGS transport bridge similar to its current routing table entry on how to get to a SNADS node that is adjacent to the PMDF-XGS transport bridge. So to add the necessary routing information on a DISOSS node, first inspect the routing table entry for how to get to a node adjacent to the PMDF-XGS transport bridge. Then add a similar entry for the PMDF-XGS transport bridge itself. For instance, if the routing table entry on a DISOSS node for a SNADS node adjacent to the PMDF-XGS transport bridge is as shown in Figure 4–4,

Figure 4–4 Existing Routing Table Entry on a DISOSS Node Not Adjacent to the PMDF-XGS Transport Bridge

```
ADD    RGN='          '  
       REN='SNADSnodeadjacenttobridge'  
       TRANSID='DSVE'  
       SSL='*'  
       QUEUE='queuename'.
```

where *SNADSnodeadjacenttobridge* is the name of the SNADS node adjacent to the PMDF-XGS transport bridge and *queuename* is the name of a queue, then you would add a routing table entry such as that shown in Figure 4–5,

Figure 4–5 New Routing Table Entry on a DISOSS Node Not Adjacent to the PMDF-XGS Transport Bridge

```
ADD    RGN='          '  
       REN='bridgeREN'  
       TRANSID='DSVE'  
       SSL='*'  
       QUEUE='queuename'.
```

using the SNADS name of the PMDF-XGS transport bridge where *bridgeREN* is shown. The effect is to instruct the SNADS node to route distributions intended for the PMDF-XGS transport bridge along the existing path to the SNADS node which has the direct link to the bridge.

PMDF-XGS

Configuring PMDF-XGS

4.6.1.3 Defining the SNA Link in VTAM and CICS on an Adjacent DISOSS Node

It is not possible to give a completely prescriptive procedure of exactly what is needed to configure the VTAM connection between the PMDF-XGS transport bridge and a directly adjacent DISOSS node, as too much depends on what is already defined. However, the PMDF-XGS transport bridge will need to be defined in VTAM, and then there needs to be a definition in CICS that refers down to this VTAM definition. Once these definitions are in place, then the DISOSS definition that refers to the CICS definition — created as shown above in Section 4.6.1.2; see in particular Figure 4–3 — completes the configuration on this adjacent DISOSS node; that DISOSS definition tells DISOSS to send distributions to the PMDF-XGS transport bridge using the CICS connection *bridgeCICSname*, where that CICS connection uses the VTAM control point *bridgeCPname* to make the actual link.

4.6.1.3.1 The VTAM Definition for the SNA Link on DISOSS

The VTAM definition will look somewhat like the one shown in Figure 4–6.

Figure 4–6 VTAM Definition of the PMDF-XGS Transport Bridge on an Adjacent DISOSS Node

```
bridgeLKname PU  ADDR=04,
                  CPNAME=bridgeCPname
                  XID=YES,RESSCB=2
bridgeCPname  LU  LOCADDR=0, DLOGMOD=#BATCH,
                  LOGAPPL=CICS
...
CICS           APPL  ACBNAME(CICS)
```

Here *bridgeLKname* is the PU (Physical Unit) name given to the PMDF-XGS transport bridge, and does not have to match any definition on the PMDF-XGS transport bridge itself *bridgeCPname* is the Local Node Name (*i.e.*, control point name) of the PMDF-XGS transport bridge. This must match the SNA VTAM node name, *i.e.*, the Local Node Name, specified in the Communications Manager definition on the PMDF-XGS transport bridge described in Section 4.6.3.1.1. (That is, note that *bridgeCPname* is the SNA VTAM layer name for the PMDF-XGS transport bridge, as contrasted to the SNA CICS layer name for the PMDF-XGS transport bridge represented by *bridgeCICSname* or as contrasted to the SNADS e-mail layer name for the PMDF-XGS transport bridge represented earlier *bridgeREN*.) *CICS* is the VTAM application name (APPL) for the CICS region where DISOSS is running, and must match the LU Name specified in the Communications Manager definition of the partner LU on the PMDF-XGS transport bridge described in Section 4.6.3.1.3. Note that the line

LOGAPPL=CICS

in the *bridgeCPname* LU (Logical Unit) definition is *required* for PMDF-XGS operation.

There can be other parameters, but these will largely be the same as for a 3270 connection from the PMDF-XGS transport bridge to the DISOSS node.

4.6.1.3.2 The CICS Definition for the SNA Link on DISOSS

For the CICS configuration both a connection definition and a session definition have to be configured in CICS. This is achieved using the CEDA commands:

```
CEDA DEFINE GROUP(group) CONNECTION(bridgeCICSname)
```

and

```
CEDA DEFINE GROUP(group) SESSION(session)
```

Here *bridgeCICSname* must match the QUEUE value specified in Section 4.6.1.2, in particular in Figure 4-3 above; *group* and *session* are locally chosen names, and do not have to match any definition on the PMDF-XGS transport bridge itself. Running these commands result in the screens shown in Figure 4-7 and Figure 4-8.

Figure 4-7 CICS CONNECTION Definition

```
CEDA DEFine
  bridgeCICSname
  Group
  Description ==> PARTNER LU
CONNECTION IDENTIFIERS
  Netname      ==> bridgeCPname
  INDSys       ==>
REMOTE ATTRIBUTES
  REMOTESystem ==>
  REMOTENAME   ==>
CONNECTION PROPERTIES
  ACessmethod ==> Vtam  Vtam|IRC|INDirect|Xm
  Protocol    ==> Appc  Appc | Lu61
  SInglesess  ==> No    Yes | No
  DATAstream ==> User  User|3270|SCs|Strf|Lms
  RECOrdformat ==> U    U | Vb
OPERATIONAL PROPERTIES
+ Autoconnect ==> No    No | Yes | All
  INService   ==> Yes   Yes | No
SECURITY
  SEcurityname ==>
  ATTachsec    ==> Local Local|Identify|Verify
                |Persistent|Hixidpe
  BINDPassword ==>      PASSWORD NOT SPECIFIED
  BINDSecurity ==> No   No | Yes
```

As described above, *bridgeCICSname* is the name CICS uses to identify the LU and *bridgeCPname* is the name VTAM uses to identify the LU and must respectively match the Local Node Name and the LU Name for the partner LU configured in Communications Manager on the PMDF-XGS transport bridge; see Section 4.6.3.1.1 and Section 4.6.3.1.3. *group* is a name chosen locally and is not significant outside of the DISOSS node.

PMDF-XGS

Configuring PMDF-XGS

Figure 4–8 CICS SESSION Definition

```
CEDA DEFine
  Sessions      : session
  Group         : group
  DEscription  ==> SESSIONS for PARTNER LU
SESSION IDENTIFIERS
  Connection    ==> bridgeCICSname
  SESSName      ==>
  NETnameq     ==>
  MODename     ==> #BATCH
SESSION PROPERTIES
  Protocol      ==> Appc          Appc | Lu61
  MAXimum      ==> 002 , 000    0-999
  RECEIVEPfx   ==>
  RECEIVECount ==>              1-999
  SENDPfx      ==>
  SENDCount    ==>              1-999
  SENDSize     ==> 1920         1-30720
  RECEIVESize  ==> 1920         1-30720
+ AUtoconnect  ==> No          No | Yes | All
  INService    ==> Yes         Yes | No
SECURITY
  SEcurityname ==>
  ATTachsec    ==> Local      Local | Identify | Verify
                                     | Persistent | Hixidpe
  BINDPassword ==>           PASSWORD NOT SPECIFIED
  BINDSecurity ==> No         No | Yes
```

Here *bridgeCICSname* is the name CICS uses to identify the LU, and must match the LU Name for the partner LU configured in Communications Manager on the PMDF-XGS transport bridge; see Section 4.6.3.1.3. *group* and *session* are names chosen locally and are not significant outside of the DISOSS node.

4.6.2 Configuring an AS/400 Node

Each SNADS node must be configured to know about the PMDF-XGS transport bridge and about every host or mail system on the PMDF side which is to be directly addressable from the SNADS side. That is, at a minimum, each SNADS node must be configured to believe that the PMDF-XGS transport bridge is another SNADS node and that the PMDF system is a SNADS node reachable through the PMDF-XGS transport bridge. Once the PMDF system is reachable from SNADS, then any host or mail system reachable by PMDF can be made reachable from SNADS via aliased or embedded addressing; however, it is usually preferable to add additional routing information to the SNADS nodes to allow direct addressing from SNADS of other commonly addressed hosts or mail systems. *The PMDF-XGS transport bridge looks to other SNADS nodes just like an AS/400 running OV/400. Other systems reachable through PMDF appear to the SNADS nodes as remote AS/400 systems.*

PMDF-XGS Configuring PMDF-XGS

The simplest way set up the SNADS nodes is to define directory information on each of the SNADS nodes so that users **ANY* at *PMDFnodeDGN*, and users **ANY* at *othersystemreachedthroughPMDF*, are all at the PMDF-XGS transport bridge (which appears to be another SNADS node), and then to define to the SNADS nodes how to send mail to the PMDF-XGS transport bridge. *PMDFnodeDGN* is technically the PMDF node's DGN, from the SNADS point of view; however, as we consider the PMDF node's REN to be the same as its DGN, it is the same value as the PMDF node's REN (and thus effectively is the PMDF node name from the SNADS point of view).

There are three parts to configuring an AS/400 node to connect to PMDF-XGS. On each AS/400 node, you must add directory information specifying that the user location for the PMDF system (and any other system reachable through PMDF that you want to add) is on the PMDF-XGS transport bridge; see Section 4.6.2.1. You must inform each AS/400 node how to route distributions to the PMDF-XGS transport bridge; see Section 4.6.2.2. And you must configure the SNA connection; see Section 4.6.2.3.

4.6.2.1 Adding Directory Information to an AS/400 Node

For an AS/400 node, user locations are defined by issuing the command WRKDIR for each SMTP node, and filling in the resulting screens as shown in Figure 4–9.

Figure 4–9 Defining SMTP Hosts to OV/400

```
ADD NEW DIRECTORY ENTRY
USER
  userid.....: *ANY
  address.....: PMDFnodeDGN
SYSTEM
  system name.....: bridgeREN
  system group.....: .....
Indirect user.....: N
Print personal mail.....: N
-----
ADD NEW DIRECTORY ENTRY
USER
  userid.....: *ANY
  address.....: anothersystemreachedthroughPMDF
SYSTEM
  system name.....: bridgeREN
  system group.....: .....
Indirect user.....: N
Print personal mail.....: N
-----
```

Figure 4–9 Cont'd on next page

PMDF-XGS

Configuring PMDF-XGS

Figure 4–9 (Cont.) Defining SMTP Hosts to OV/400

```

ADD NEW DIRECTORY ENTRY

USER
  userid.....: *ANY
  address.....: yetanotherssystemreachedthroughPMDF
SYSTEM
  system name.....: bridgeREN
  system group.....: .....
Indirect user.....: N
Print personal mail.....: N
-----
...

```

4.6.2.2 Adding Routing Information on an AS/400 Node

Each AS/400 node needs to be told how to route messages to the PMDF-XGS transport bridge. Exactly where a particular AS/400 node should route a distribution eventually intended for the PMDF-XGS transport bridge will depend upon the topology of your SNADS network. The SNADS node directly adjacent to the PMDF-XGS transfer system will need a new entry pointing to the PMDF-XGS transport bridge; a SNADS node which is not adjacent to the PMDF-XGS transport bridge, but which presumably does already have an entry specifying how to get to the SNADS node which is adjacent to the PMDF-XGS transport bridge, will also need a new entry for the PMDF-XGS transport bridge similar to its current entry for a SNADS node which is adjacent to the PMDF-XGS transport bridge.

Figure 4–10 Defining the Route to the OS/2 Transport Bridge on an Adjacent AS/400 Node

```

ROUTING TABLE ENTRY

Destination system
  Name / Group.....: bridgeREN
Description.....: PMDF-XGS OS/2 transport bridge
Service level
  Fast:
    Queue name.....: queuename
    Maximum hops.....: *DFT
  Status:
    Queue name.....: queuename
    Maximum hops.....: *DFT
  Data high:
    Queue name.....: queuename
    Maximum hops.....: *DFT
  Data low:
    Queue name.....: queuename
    Maximum hops.....: *DFT

```

PMDF-XGS Configuring PMDF-XGS

If the AS/400 node is adjacent to the PMDF-XGS transport bridge, then it will need a new routing table entry along the lines of Figure 4–10. Here *bridgeREN* is the name of the PMDF-XGS transport bridge and where *queuename* is the queue name you will use for the SNA link, as described in Section 4.6.2.3.

If the AS/400 node is not adjacent to the PMDF-XGS transport bridge, then it will need a new routing table entry on how to get to the PMDF-XGS transport bridge similar to its current routing table entry on how to get to a SNADS node that is adjacent to the PMDF-XGS transport bridge. So to add the necessary routing information on an AS/400 node, first inspect the routing table entry for how to get to a node adjacent to the PMDF-XGS transport bridge. Then add a similar entry for the PMDF-XGS transport bridge itself.

For instance, if the routing table entry on an AS/400 node on how to get to a SNADS node adjacent to the PMDF-XGS transport bridge is as shown in Figure 4–11,

Figure 4–11 Existing Routing Table Entry on an AS/400 Node Not Adjacent to the PMDF-XGS Transport Bridge

```
ROUTING TABLE ENTRY

Destination system
  Name / Group.....:  SNADSnodeadjacenttobridge
Description.....:  node adjacent to the PMDF-XGS OS/2 system
Service level
  Fast:
    Queue name.....:  queuename
    Maximum hops.....:  *DFT
  Status:
    Queue name.....:  queuename
    Maximum hops.....:  *DFT
  Data high:
    Queue name.....:  queuename
    Maximum hops.....:  *DFT
  Data low:
    Queue name.....:  queuename
    Maximum hops.....:  *DFT
```

where *SNADSnodeadjacenttobridge* is the name of a SNADS node which is adjacent to the PMDF-XGS transport bridge and *queuename* is the name of the queue for that connection, then you will need to add a similar routing table entry for the PMDF-XGS transport bridge. This new routing information for the PMDF-XGS PMDF-XGS transport bridge can be added using the command CFGDSTSVR, selecting “Routing Table Entry” and filling in the information as in Figure 4–12. Here *bridgeREN* is the name of the PMDF-XGS transport bridge and *queuename* should be the same queue as used in Figure 4–11. Note that you are not defining a new route; just telling the AS/400 system to use the same route for mail bound for the PMDF-XGS transport bridge as is used for mail bound for the SNADS node adjacent to the PMDF-XGS transport bridge.

PMDF-XGS

Configuring PMDF-XGS

Figure 4–12 New Routing Table Entry on an AS/400 Node Not Adjacent to the PMDF-XGS Transfer System

```
ROUTING TABLE ENTRY

Destination system
  Name / Group.....:  bridgeREN
Description.....:    PMDF-XGS OS/2 transport bridge
Service level
  Fast:
    Queue name.....:  queuename
    Maximum hops.....: *DFT
  Status:
    Queue name.....:  queuename
    Maximum hops.....: *DFT
  Data high:
    Queue name.....:  queuename
    Maximum hops.....: *DFT
  Data low:
    Queue name.....:  queuename
    Maximum hops.....: *DFT
```

4.6.2.3 Defining the SNA Link on an Adjacent AS/400 System

On an AS/400 system which is adjacent to the PMDF-XGS transport bridge, the SNA link to the PMDF-XGS transport bridge must be defined. Such a link is exactly the same as is required for a 5250 terminal session. So if you normally define 5250 sessions statically on the AS/400, then you can define the PMDF-XGS transport bridge exactly as you would any terminal controller or OS/2 based emulator. If you normally allow your 5250 sessions to autoconfigure, then no steps are required to configure the SNA connection to the PMDF-XGS transport bridge. In either case you have to configure the AS/400 so that it can send the distributions bound for the PMDF-XGS transport bridge over the right SNA session. This is done using the command CFGDSTSRV, and selecting “Routing Table Entry” and “Distribution Queue Entry”.

The routing table entry step can have been performed earlier, as described in Section 4.6.2.2; see in particular Figure 4–10. The routing table entry tells SNADS that distributions bound for a particular node are to be put in a particular queue, and then the distribution queue entry tells SNADS how to send the distribution entries from a queue. So note that the queue name in the routing table entry must match the queue name in the distribution queue entry. Apart from that, the precise queue name used is not significant outside the AS/400, and it is common to use the same name for the queue as for the node at the other end of the link.

Figure 4–13 shows a distribution queue entry for an AS/400 node adjacent to the PMDF-XGS transport bridge; such an entry can be defined using the command CFGDSTSRV and selecting “Distribution Queue Entry”.

Figure 4–13 Defining the AS/400 Distribution Queue for the OS/2 Transfer System

```

DISTRIBUTION QUEUE
Queue name.....:      queuename
Queue type.....:      *SNADS
Remote location name.....:  bridgeCPname
Mode name.....:      #BATCH
Remote net ID.....:     *LOCATTR
Local location name.....:  *LOCATTR
Normal priority
  Send time:
    From/to.....:      00:00   23:59
    Force.....:       :       :
  Send depth.....:     1
High priority
  Send time:
    From/to.....:      00:00   23:59
    Force.....:       :       :
  Send depth.....:     1
Normal priority
  Send time:
    From/to.....:      00:00   23:59
    Force.....:       :       :
  Send depth.....:     1

```

Here *queuename* is the name chosen for the distribution queue, matching that used in the routing table entry, and where *bridgeCPname* is the SNA name of the PMDF-XGS transport bridge, *i.e.*, the Local Node Name configured in Communications Manager on the PMDF-XGS transport bridge system; see Section 4.6.3.1.1.

4.6.3 Configuring the PMDF-XGS Transport Bridge

There are a number of aspects to configuring the PMDF-XGS transport bridge: configuring the SNA connection to the adjacent SNADS node(s); configuring the transfer programs to run on the transport bridge; and if using an OS/2 system as the transport bridge (NT systems normally have TCP/IP already configured), configuring the TCP/IP connection to the PMDF/PMDF-XGS system. The following sections describe the steps involved.

4.6.3.1 Configuring the SNA Connection on the PMDF-XGS Transport Bridge

Configuring the SNA connection from the PMDF-XGS transport bridge to the adjacent SNADS node is performed using Personal Communications on an NT system, or using Communications Manager on an OS/2 system. After gathering the necessary information, the configuration of the SNA parameters using Personal Communications or Communications Manager will comprise four or five steps:

- *The local node definition.* The local node, or *control point*, is the SNA name of the OS/2 transport bridge, and a type 6.2 LU. (As the control point is a type 6.2 LU, note that this is the only local resource that needs to be defined.)

PMDF-XGS

Configuring PMDF-XGS

- *The SNA connection definition, i.e., the definition of the link to the immediately adjacent SNADS node.* The link defines how to find the immediately adjacent SNADS node; for instance, by LAN address.
- *The partner LU definition for the immediately adjacent SNADS node.* The partner LU is the host APPL (CICS for DISOSS), or location name for the AS/400.
- *The CPI-C side information (i.e., symbolic destination) definition.* The symbolic destination is a name used by the gateway to refer to the set of values needed to make the outbound connection.
- *The transaction program defaults settings.* On an OS/2 transport bridge system, although not strictly required for PMDF-XGS transport bridge operation, changing certain default settings is strongly recommended.

More details on the SNA parameters for each step, set using Personal Communications (NT) or Communications Manager (OS/2), which are critical for use with PMDF-XGS are described in Section 4.6.3.1.1, Section 4.6.3.1.2, Section 4.6.3.1.3, Section 4.6.3.1.4, and Section 4.6.3.1.5, respectively, below.

On OS/2, after configuring the SNA connection for the PMDF-XGS transport bridge system using Communications Manager, you should have a Node Definition File, (a `.ndf` file), in Communications Manager's `cmlib` directory which should be similar to that shown in Figure 4–14; you can want to look over your Node Definition File for comparison.

On NT, there is no text file version of the configuration (no equivalent of the Node Definition File on OS/2).

Finally, after configuring your SNA connection, make sure to shadow the appropriate icon or icons to your system's startup folder: on OS/2 shadow Communications Manager's "Start Communications" icon to your system's startup folder; on NT shadow its equivalent icons to your system's startup folder.

4.6.3.1.1 Parameters for the Local Node Definition

Network ID

All nodes and LU's (Logical Units) have what is called a fully qualified name. This consists of a network id and a name. Usually all nodes within a network have the same network name, generally consisting of two characters representing the country followed by three characters representing the organization. The PMDF-XGS transport bridge should use the same network name as the immediately adjacent SNADS node, that is, the SNADS node to which it is connecting. If this immediately adjacent SNADS node is a mainframe, *i.e.*, a DISOSS system, then the network id will be defined in the VTAM start parameter list (typically member name `ATCSTR00` in `SYS1.VTAMLIST`). The VTAM keyword for this parameter is `NETID=`. If the immediately adjacent SNADS node is an AS/400, then the network name is the parameter `LCLNETID`, and can be found using the `DSPNETA` command. It is a name of up to 8 characters, upper case alphanumeric or #, @ or \$. The first character must not be a digit. This parameter corresponds to the field marked ❶ in Figure 4–14.

Local Node Name

This forms the second part of the fully qualified node name. The local node is the PU (Physical Unit), and a type 6.2 LU (Logical Unit) as well. This must match the CPNAME= parameter in the PU definition of VTAM (if there is one), and the label on the LU definition. On the AS/400 this must match the controller name and the device name defined for this node. The node name can have up to 8 characters, upper case alphanumeric or #, @ or \$. The first character must not be a digit. This parameter corresponds to the field marked ② in Figure 4–14.

Local Node ID

In some VTAM configurations, nodes are recognised by number instead of name. In this case the parameters IDBLK= and IDNUM= will appear on the PU definition in the VTAM listing. If these parameters are present, then the three hexadecimal digit IDBLK value and the five digit IDNUM value make up the Node ID and must be specified in the Communications Manager definition. This parameter corresponds to the field marked ⑥ in Figure 4–14.

4.6.3.1.2 Parameters for the SNA Connection Definition

LAN Destination Address

Whatever type of connection you are using between the PMDF-XGS transport bridge and the adjacent SNADS node, you have to tell the Communications Manager how to find the adjacent node. Typically you will be using a LAN connection, in which case the LAN address to which you connect to reach the remote machine has to be provided. This can be the address of the FEP (Front End Processor) or the AS/400, but in more complicated networks it could be the address of some other communications controller. This parameter corresponds to the field marked ⑤ in Figure 4–14.

Partner Network ID

This will usually be the same as the local node Network ID, described above in Section 4.6.3.1.1. This parameter corresponds to the field marked ⑦ in Figure 4–14.

Partner Node Name

This is the name of the remote node. If the remote node is running VTAM, then this is the SSCP (System Services Control Point) name of VTAM. This can be found in the start parameter list with the keyword SSCPNAME=. If the remote node is an AS/400, then this is the Control Point Name (LOCCPNAME) of the AS/400, and can be found by using the DSPNETA command. This parameter corresponds to the field marked ③ in Figure 4–14.

4.6.3.1.3 Parameters for the Partner LU Definition

Network ID

This will be the same as the local node Network ID. This parameter corresponds to the field marked ⑬ in Figure 4–14.

LU Name

If the remote node is running VTAM, then this is the APPL id of CICS. If the remote node is an AS/400, then this will be the Default Local Location Name (LCLLOCNAME) of the AS/400, and can be found by using the DSPNETA command. This is a name of up to four characters, and must match the value specified where CICS is shown in Figure 4–6. This parameter corresponds to the field marked ⑭ in Figure 4–14.

PMDF-XGS

Configuring PMDF-XGS

Alias

The Alias for the partner LU has no external significance in the SNA network, but it is very significant for the PMDF side as it determines which SNADS channel PMDF will use for incoming connections from that node. When using a single snads_local channel to connect to the SNADS world, specify this parameter as LOCAL, or as a name that starts with a non-alphabetic character. In a configuration with multiple SNADS channels in use, each additional channel, say snads_nodex, would require an additional partner LU definition with this parameter specified as NODEX.) This is a case sensitive field; for PMDF-XGS operation, this *must be entered in upper case*. This parameter corresponds to the field marked ⑮ in Figure 4–14.

4.6.3.1.4 Parameters for the CPI-C Side Information (symbolic destination)

Symbolic Destination Name

This must match the Alias for the partner LU, described above in Section 4.6.3.1.3. This is a case sensitive field; for PMDF-XGS operation, this *must be entered in upper case*. This parameter corresponds to the field marked ⑮ in Figure 4–14.

Partner LU Fully Qualified Name

This is the Network ID and LU Name of the partner LU, discussed in Section 4.6.3.1.3. The parameter corresponds to the field marked ⑯ in Figure 4–14.

Partner TP

This is a service TP. On an NT transport bridge, it must be set to the value 21002. On an OS/2 transport bridge, it must be set to the value X'21'002. The parameter corresponds to the field marked ⑰ in Figure 4–14. Note also that the Service TP option must be set.

Security

This must be set to NONE. This lack of a value for this parameter corresponds to the location marked ⑱ in Figure 4–14.

Mode Name

You should generally use the mode #BATCH for SNADS connections. The mode name #BATCH is predefined in VTAM and in the AS/400. (Conceivably you could define and use some other mode of similar priority that, for instance, would not interfere with interactive terminal sessions.) This parameter corresponds to the field marked ⑳ in Figure 4–14.

4.6.3.1.5 Parameters for the Transaction Program Defaults Definition on OS/2

On an OS/2 transport bridge, most default values for the transaction program defaults are suitable for the PMDF-XGS transport bridge SNA connection, with the following exception:

Directory For Inbound Attaches

This should be set to nothing; *i.e.*, delete the regular default * value in this field. This lack of a value for this parameter corresponds to the location marked ㉑ in Figure 4–14.

4.6.3.1.6 OS/2 Transport Bridge Node Definition File Format

On OS/2, the SNA configuration created using Communications Manager has a corresponding text file, the node definition file. Figure 4–14 shows a sample of the format of such a node definition file.

Figure 4–14 The NDF File on the Transport Bridge System

```

DEFINE_LOCAL_CP  FQ_CP_NAME(NetworkID.LocalNodeName)  ① ②
                  CP_ALIAS(LocalNodeAlias)  ③
                  NAU_ADDRESS(INDEPENDENT_LU)  ④
                  NODE_TYPE(EN)  ⑤
                  NODE_ID(X'LocalNodeID')  ⑥
                  NW_FP_SUPPORT(NONE)
                  HOST_FP_SUPPORT(YES)
                  MAX_COMP_LEVEL(NONE)
                  MAX_COMP_TOKENS(0);

DEFINE_LOGICAL_LINK  LINK_NAME(LINK0001)  ⑦ ⑧
                     FQ_ADJACENT_CP_NAME(PartnerNetworkID.PartnerNodeName)
                     ADJACENT_NODE_TYPE(LEARN)
                     DLC_NAME(IBMTRNET)
                     ADAPTER_NUMBER(0)
                     DESTINATION_ADDRESS(X'LANDestinationAddress')  ⑨
                     ETHERNET_FORMAT(NO)  ⑩
                     CP_CP_SESSION_SUPPORT(NO)  ⑪
                     SOLICIT_SSCP_SESSION(NO)  ⑫
                     ACTIVATE_AT_STARTUP(YES)
                     USE_PUNAME_AS_CPNAME(NO)
                     LIMITED_RESOURCE(USE_ADAPTER_DEFINITION)
                     LINK_STATION_ROLE(USE_ADAPTER_DEFINITION)
                     MAX_ACTIVATION_ATTEMPTS(USE_ADAPTER_DEFINITION)
                     EFFECTIVE_CAPACITY(USE_ADAPTER_DEFINITION)
                     COST_PER_CONNECT_TIME(USE_ADAPTER_DEFINITION)
                     COST_PER_BYTE(USE_ADAPTER_DEFINITION)
                     SECURITY(USE_ADAPTER_DEFINITION)
                     PROPAGATION_DELAY(USE_ADAPTER_DEFINITION)
                     USER_DEFINED_1(USE_ADAPTER_DEFINITION)
                     USER_DEFINED_2(USE_ADAPTER_DEFINITION)
                     USER_DEFINED_3(USE_ADAPTER_DEFINITION);

DEFINE_PARTNER_LU  FQ_PARTNER_LU_NAME(NetworkID.LUName)  ⑬ ⑭
                   PARTNER_LU_ALIAS(Alias)  ⑮
                   PARTNER_LU_UNINTERPRETED_NAME(LUName)  ⑯
                   MAX_MC_LL_SEND_SIZE(32767)
                   CONV_SECURITY_VERIFICATION(NO)
                   PARALLEL_SESSION_SUPPORT(YES);

DEFINE_PARTNER_LU_LOCATION  FQ_PARTNER_LU_NAME(NetworkID.LUName)  ⑬ ⑭
                             WILDCARD_ENTRY(NO)  ⑦ ⑧
                             FQ_OWNING_CP_NAME(PartnerNetworkID.PartnerNodeName)
                             LOCAL_NODE_NN_SERVER(NO);

```

Figure 4–14 Cont'd on next page

PMDF-XGS

Configuring PMDF-XGS

Figure 4–14 (Cont.) The NDF File on the Transport Bridge System

```
DEFINE_DEFAULTS  IMPLICIT_INBOUND_PLU_SUPPORT(YES)
                  DEFAULT_MODE_NAME(BLANK)
                  MAX_MC_LL_SEND_SIZE(32767) ⑰
                  DEFAULT_TP_OPERATION(NONQUEUED_AM_STARTED)
                  DEFAULT_TP_PROGRAM_TYPE(BACKGROUND)
                  DEFAULT_TP_CONV_SECURITY_RQD(NO)
                  MAX_HELD_ALERTS(10);

DEFINE_CPIC_SIDE_INFO SYMBOLIC_DESTINATION_NAME(SymbolicDestinationName) ⑱
                      FQ_PARTNER_LU_NAME(NetworkID.LUName) ⑬ ⑭
                      MODE_NAME(ModeName) ⑲
                      SNA_SERVICE_TP_NAME(X'21',002); ⑳ ㉑

START_ATTACH_MANAGER;
```

- ① The *NetworkID* value corresponds to the Network ID field described in Section 4.6.3.1.1.
- ② The *LocalNodeName* value corresponds to the Local Node Name field described in Section 4.6.3.1.1.
- ③ The *LocalNodeAlias* does not matter to PMDF-XGS.
- ④ The value in the `NAU_ADDRESS` field should essentially always be `INDEPENDENT_LU`, as shown.
- ⑤ The value in the `NODE_TYPE` field depends upon what sort of node this is. With PMDF-XGS, the bridge transport node is essentially always an end node, corresponding to the value shown here, `EN`.
- ⑥ The *LocalNodeID* value corresponds to the Local Node ID field described in Section 4.6.3.1.1.
- ⑦ The *PartnerNetworkID* value corresponds to the Partner Network ID field described in Section 4.6.3.1.2.
- ⑧ The *PartnerNodeName* value corresponds to the Partner Node Name field described in Section 4.6.3.1.2.
- ⑨ The first twelve characters of the *LANDestinationAddress* value correspond to the Lan Destination Address field described in Section 4.6.3.1.2; the last two characters in this value are the SAP address, which is usually 04.
- ⑩ The correct value for a site's `ETHERNET_FORMAT` field depends upon the sort of network in use, Ethernet or token ring. The value shown here, `NO`, would be correct for a token ring network.
- ⑪ The correct value for a site's `CP_CP_SESSION_SUPPORT` parameter depends upon whether the link is end node to end node, or end node to host, or end node to network. The value shown here would be correct for an end node to end node link.
- ⑫ The correct value for a site's `SOLICIT_SSCP_SESSION` parameter depends upon whether the link is to support dependent LU's, such as a 3270 session. The value shown here, `NO`, implies no dependent LU support.

PMDF-XGS Configuring PMDF-XGS

- ⑬ The *NetworkID* value corresponds to the Network ID field for the partner LU described in Section 4.6.3.1.3.
- ⑭ The *LUName* value corresponds to the LU Name field described in Section 4.6.3.1.3.
- ⑮ The *Alias* value corresponds to the Alias field described in Section 4.6.3.1.3; for the case of a single snads_local channel, this parameter value will be LOCAL or a value starting with a non-alphabetic character.
- ⑯ The Uninterpreted Name is almost always set to the same value as the LU Name for the partner LU; that is, this is almost always the same as ⑭.
- ⑰ Note that there is no DIRECTORY_FOR_INBOUND_ATTACHES field in this section, since the default * value was deleted.
- ⑱ The *SymbolicDestinationName* value corresponds to the Partner Node Name field described in Section 4.6.3.1.4. This value should match the value shown in ⑮.
- ⑲ The *ModeName* value corresponds to the Mode Name field described in Section 4.6.3.1.4.
- ⑳ Note that the SNA_SERVICE_TP_NAME field is set to X'21',002 as required for the Partner TP field described in Section 4.6.3.1.4. (During the configuration with Communications Manager, this value the value should be entered without a comma; the comma will then appear in the actual file.)
- ㉑ Note that there is no SECURITY field, as required and described in Section 4.6.3.1.4.

4.6.3.2 Configuring the Programs on the NT PMDF-XGS Transport Bridge

This section describes configuring the PMDF-XGS transport bridge programs on an NT transport bridge.

In the directory on the transport bridge system into which the PMDF-XGS transport bridge programs were installed, typically `c:\snadsrv`, there is a file `xgs.cmd` created according to your answers during the (configuration portion of the) installation of the PMDF-XGS transport bridge programs. This is a command file, similar to a DOS `.BAT` file, which starts three components of PMDF-XGS which must run on the PMDF-XGS transport bridge. These components are:

- `sendsrv`, which accepts SNADS distributions outbound from PMDF and sends them out into the SNADS network; and
- `recvsrv`, which accepts SNADS distributions from the SNADS network and sends them to the PMDF system.

Some of these components require parameters to work properly. The installation procedure itself will take you through a configuration dialogue, creating an `xgs.cmd` procedure with parameters in accordance with your answers. See Section 4.7, step 10, for a sample installation and configuration of these programs on the PMDF-XGS transport bridge system. Section 4.6.3.2.1 below discusses the individual parameters and their uses.

Once the `xgs.cmd` procedure has been created, be sure to shadow the `xgs.cmd` icon to the system's "Startup" folder so that it will be autostarted.

PMDF-XGS

Configuring PMDF-XGS

4.6.3.2.1 Parameters for the NT PMDF-XGS Transport Bridge Programs

`sendsrv` takes up to two parameters:

```
sendsrv [bridgeport] [loghost]
```

The *bridgeport* parameter is the TCP/IP port number on which `sendsrv` accepts connections. If no parameters are given, this defaults to 9994. The *loghost* parameter is the TCP/IP host name to whose syslog daemon logging output is to be sent. If this parameter is not specified, it defaults to `localhost`.

`recvsrv` has four required parameters as well as one additional optional parameter:

```
recvsrv PMDFhost dispatcherport bridgeREN docsize [loghost]
```

The *PMDFhost* parameter is the TCP/IP host name of the PMDF node running the rest of the PMDF-XGS code. The *dispatcherport* parameter is the TCP/IP port number on which the PMDF Service Dispatcher on the PMDF system listens for requests from the PMDF-XGS receive server on the PMDF-XGS transport bridge. 9993 is the usual number. The *bridgeREN* parameter is the SNADS name of the PMDF-XGS transport bridge, *i.e.*, that name specified as the Local Node Name in the Communications Manager configuration on the PMDF-XGS transport bridge and specified as the `BRIDGE_REN` option value in the channel option file on the PMDF system. This is used in status distributions returned from PMDF-XGS. The *docsize* parameter is the maximum size of SNADS distributions that the PMDF-XGS transport bridge is allowed to receive. There is no reason to be too conservative here: a number like 10000000 is probably reasonable. The optional *loghost* parameter is the TCP/IP host name to whose syslog daemon to send logging output. If this parameter is not specified, it defaults to `localhost`.

Example 4–1 shows an example `xgs.cmd` file.

Example 4–1 Sample `xgs.cmd` File

```
start sendsrv 9994 localhost
start recvsrv naples.example.com 9993 BRIDGE 10000000 localhost
```

4.6.3.3 Configuring the Programs on the OS/2 PMDF-XGS Transport Bridge

In the directory on the transport bridge system into which the PMDF-XGS transport bridge programs were installed, typically `c:\snadsrv`, there is a file `xgs.cmd` created according to your answers during the (configuration portion of the) installation of the PMDF-XGS transport bridge programs. This is a command file, similar to a DOS `.BAT` file, which starts three components of PMDF-XGS which must run on the PMDF-XGS transport bridge. These components are:

- `xcontrol`, which manages the SNMP requests;
- `sendsrv`, which accepts SNADS distributions outbound from PMDF and sends them out into the SNADS network; and

PMDF-XGS Configuring PMDF-XGS

- `recvsrv`, which accepts SNADS distributions from the SNADS network and sends them to the PMDF system.

Some of these components require parameters to work properly. The installation procedure itself will take you through a configuration dialogue, creating an `xgs.cmd` procedure with parameters in accordance with your answers. See Section 4.7, step 10, for a sample installation and configuration of these programs on the PMDF-XGS transport bridge system. Section 4.6.3.3.1 below discusses the individual parameters and their uses.

Once the `xgs.cmd` procedure has been created, be sure to shadow the `xgs.cmd` icon to the system's "Startup" folder so that it will be autostarted.

4.6.3.3.1 Parameters for the OS/2 PMDF-XGS Transport Bridge Programs

`xcontrol` takes up to two parameters:

```
xcontrol [SNMPcommunity] [SNMPport]
```

The `SNMPcommunity` parameter is the SNMP community name which `xcontrol` uses to interact with the SNMP daemon. If no parameter is provided, the name `public`, is assumed, which is the default community name defined in the daemon. The `SNMPport` parameter is the port number used by `sendsrv` and `recvsrv` to interact with `xcontrol`; it defaults to 9990.

`sendsrv` takes up to four parameters:

```
sendsrv [bridgeport] [loghost] [SNMPport] [directory]
```

The `bridgeport` parameter is the TCP/IP port number on which `sendsrv` accepts connections. If no parameters are given, this defaults to 9994. The `loghost` parameter is the TCP/IP host name to whose syslog daemon logging output is to be sent. If this parameter is not specified, it defaults to `localhost`. The `SNMPport` parameter is the port with which to interact with the SNMP daemon. If this parameter is not specified, it defaults to 9990. The `directory` parameter is the SNADS capture directory. It is possible to ask the PMDF-XGS transport bridge to keep copies of all, all failing, or none of, the SNADS distributions it sends. If the `directory` parameter is specified, then the captured files are placed in that directory. The default is to put the captured files in the current directory.

`recvsrv` has four required parameters as well as three additional optional parameters:

```
recvsrv PMDFhost dispatcherport bridgeREN docsize [loghost] [SNMPport] [directory]
```

The `PMDFhost` parameter is the TCP/IP host name of the PMDF node running the rest of the PMDF-XGS code. The `dispatcherport` parameter is the TCP/IP port number on which the PMDF Service Dispatcher on the PMDF system listens for requests from the PMDF-XGS receive server on the PMDF-XGS transport bridge. 9993 is the usual number. The `bridgeREN` parameter is the SNADS name of the PMDF-XGS transport bridge, *i.e.*, that name specified as the Local Node Name in the Communications Manager configuration on the PMDF-XGS transport bridge and specified as the `BRIDGE_REN` option value in the channel option file on the PMDF system. This is used in status

PMDF-XGS

Configuring PMDF-XGS

distributions returned from PMDF-XGS. The *docsize* parameter is the maximum size of SNADS distributions that the PMDF-XGS transport bridge is allowed to receive. There is no reason to be too conservative here: a number like 10000000 is probably reasonable. The optional *loghost* parameter is the TCP/IP host name to whose syslog daemon to send logging output. If this parameter is not specified, it defaults to *localhost*. The optional *SNMPport* parameter is the port with which to interact with the SNMP daemon. If this parameter is not specified, it defaults to 9990. The optional *directory* parameter specifies a capture directory. If this optional parameter is specified, then the PMDF-XGS transport bridge writes a copy of each outbound SNADS distributions to the specified directory.

Example 4–2 shows an example *xgs.cmd* file.

Example 4–2 Sample *xgs.cmd* File

```
start xcontrol
start sendsrv 9994 localhost 9990 c:/capture
start recvsrv naples.example.com 9993 BRIDGE 10000000 localhost 9990
c:/capture
```

4.6.3.4 Configuring TCP/IP on an OS/2 PMDF-XGS Transport Bridge

Configuring TCP/IP under OS/2 is very similar to configuring TCP/IP under UNIX: it is largely configured by editing the files *hosts*, and *resolv*, *etc.*, using the IBM supplied “TCP/IP configure” program. These files are to be found in the directory *\tcpip\etc* if you are using TCPIP/2 v2.0, or in *\mptn\etc* if you are using Warp Connect.

There are, however a couple of points of which to be aware. You must make sure that the loopback driver is started, and that the hostname *localhost* is defined. The loopback driver is started by the line

```
ifconfig lo 127.0.0.1
```

in the file *setup.cmd* in *\tcpip\bin* (TCPIP/2 v2.0) or in *\mpts\bin* (Warp Connect). The hostname *localhost* must be defined in the *hosts* file in *\tcpip\etc* (TCPIP/2 v2.0) or in *\mpts\etc* (Warp Connect) as

```
127.0.0.1 localhost
```

The transport bridge component of the PMDF-XGS gateway collects statistics which can be queried from an SNMP network management center. It does this through the *dpi* interface of the SNMP daemon. This must be configured to start, and to support the *dpi* interface. This is most easily configured through the configuration notebook. (If you are using Warp Connect, the parameters *-dpi shm -dpi tcp* must be specified.) Configuring this through the configuration notebook will result in the line

```
start /min snmpd
```

being added to the file *tcpstart.cmd* in the directory *\tcpip\bin* for TCPIP/2 v2, or

```
start /min snmpd -dpi shm -dpi tcp
```

being added to the file `tcpstart.cmd` in the directory `\mptn\bin` for Warp Connect.

4.6.4 Configuring PMDF-XGS on the PMDF System

PMDF-XGS on the PMDF system should be configured using the PMDF-XGS configuration utility, PMDF CONFIGURE XGS (OpenVMS) or `pmdf configure xgs` (UNIX). This utility will, when it is finished, produce a checklist of final steps you must manually perform in order to complete your configuration. The following subsections document these steps.

4.6.4.1 Adding the Channel to the Configuration File

The PMDF-XGS configuration utility creates a `snads_local` channel for you; you do not need to undertake the steps described in this section. This channel and associated rewrite rules appear in the `xgs.chans` and `xgs.rules` files in the PMDF table directory.

At least one PMDF-XGS channel is required to connect to the PMDF-XGS OS/2 transport bridge and through it to the SNADS world. The entry for the channel should look something like:

```
snads_local defragment 733 header_733 maxheaderaddrs 1 addrspersfile 256 \  
  charset7 US-ASCII charset8 ISO-8859-1  
SNADS-DAEMON SNADS-name-for-PMDF-system
```

Here *SNADS-name-for-PMDF-system* should be the name by which the SNADS world knows the PMDF system.

In order to allow sending from the PMDF side to SNADS users using

```
username@snadsnode.yourdomain
```

style addresses, you will want a number of rewrite rules associating the various SNADS nodes and any corresponding SNADS pseudodomain names with this channel, *i.e.*,

```
snadsnode1           $U%snadsnode1.yourdomain  
snadsnode1.yourdomain $U%snadsnode1.yourdomain@SNADS-DAEMON  
snadsnode2           $U%snadsnode2.yourdomain  
snadsnode2.yourdomain $U%snadsnode2.yourdomain@SNADS-DAEMON  
...
```

The above style of rewrite rules are suitable if you are using pseudodomain names for your SNADS nodes that have the actual SNADS node name as the first (host) part of the pseudodomain name. If you use pseudodomain names that do *not* use the actual SNADS node name as the first (host) part of the pseudodomain name, then you will want rewrite rules that rewrite a SNADS pseudodomain name to the corresponding actual SNADS node name when going to SNADS, and that rewrite an actual SNADS node name to the corresponding SNADS pseudodomain name when coming from SNADS, *i.e.*,

PMDF-XGS

Configuring PMDF-XGS

```
snadsnode           $U%snadspseudodomainname
snadspseudodomainname $U%snadsnode@SNADS-DAEMON$E$F
snadspseudodomainname $U%snadsnode@SNADS-DAEMON$Qsnads_local
snadspseudodomainname $U%snadspseudodomainname@SNADS-DAEMON
```

In order to allow SNADS users to send to the PMDF node and specified other systems on the PMDF side using

username@PMDFnode

style addresses, you will want rewrite rules associating the short form names that are all that SNADS users can enter with the real domain names on the PMDF side, *i.e.*,

```
PMDFnode           $U%PMDFdomainname$Msnads_local
othernodeonPMDFside1 $U%otherdomain1$Msnads_local
othernodeonPMDFside2 $U%otherdomain2$Msnads_local
...
```

Note that these are source channel specific rewrite rules (the `$Msnads_local` clause) so that these rewrite rules apply only for messages originating from the `snads_local` channel.

4.6.4.2 Channel Option File

The PMDF-XGS configuration utility creates a minimal `snads_local_option` file for you in the PMDF table directory.

PMDF-XGS requires a channel-specific option file to specify various configuration options. This file is read by the PMDF-XGS channel programs during initialization. The names of the mandatory options are

- BRIDGE_HOST,
- BRIDGE_PORT, and
- BRIDGE_REN.

Their significance and usage are described below in Section 4.6.4.2.3. Before setting up the PMDF-XGS option file, you must know the correct values to specify for these mandatory options.

4.6.4.2.1 Location of the Option File

PMDF-XGS option files are stored in the PMDF table directory and must have names of the form *channelname_option* with *channelname* the name of the PMDF-XGS channel to which this option file applies. Since the channel name for PMDF-XGS is usually `snads_local`, the filename is usually `PMDF_TABLE:snads_local_option`. (OpenVMS) or `/pmdf/table/snads_local_option` (UNIX).

4.6.4.2.2 Format of the Option File

Option files consist of several lines. Each line contains the setting for one option. An option setting has the form:

option=value

value can be either a string or an integer, depending on the option's requirements. If the option accepts an integer value, a base can be specified using notation of the form *b%v*, where *b* is the base expressed in base 10 and *v* is the actual value expressed in base *b*.

Comments are allowed. Any line that begins with an exclamation point is considered to be a comment and is ignored. Blank lines are also ignored in any option file.

4.6.4.2.3 Contents of the Option File

The available options are:

ADJACENT_NODE_REN (string)

This specifies the REN of the SNADS node immediately adjacent to the PMDF-XGS transport bridge.

ADJACENT_NODE_RGN (string)

This specifies the RGN of the SNADS node immediately adjacent to the PMDF-XGS transport bridge. If a value is specified for this option, then a value for the ADJACENT_NODE_REN option must also be specified.

BRIDGE_HOST (string)

This is the TCP/IP hostname of the PMDF-XGS transport bridge.

BRIDGE_PORT (integer)

This is the port on which the PMDF-XGS send process on the PMDF-XGS transport bridge listens for communications from the PMDF system, (typically 9994). This must match the value of the *bridgeport* parameter on the *sendsrv* command line in the *xgs.cmd* file on the PMDF-XGS transport bridge.

BRIDGE_REN (string)

This is the SNADS name (the REN) of the PMDF-XGS transport bridge. This must match the name that the SNADS nodes will be using for the PMDF-XGS transport bridge, as configured in the directory and routing table entries on the SNADS nodes.

SAVE_HEADERS (0 or 1)

The *SAVE_HEADERS* option is used to control whether or not RFC 822 headers are retained as additional text in SNADS messages produced by PMDF. A value of 0 is the default, and specifies that no headers are to be retained. A value of 1 places the RFC 822 headers as additional text at the bottom of the message.

SUPPRESS_DELIVERY_RECEIPTS (integer)

SUPPRESS_READ_RECEIPTS (integer)

These options control whether delivery and read receipt requests are passed from PMDF to SNADS and from SNADS to PMDF; they take bit encoded integer values. Bit 0 controls the PMDF to SNADS direction; setting this bit causes PMDF to suppress the respective sort of receipt request going to SNADS. Bit 1 controls the SNADS to PMDF direction;

PMDF-XGS

Configuring PMDF-XGS

setting this bit causes PMDF to suppress the respective sort of receipt request coming from SNADS. The default value for each is 0, meaning that receipt requests are passed in each direction.

Note that some SNADS implementations have idiosyncratic interpretations of the COD (Confirmation_Of_Delivery) and RRR (Read_Receipt_Requested) bits, and can, for instance, unconditionally generate a delivery receipt if a read receipt is requested.

TRANSACTIONS_PER_CONVERSATION (integer)

This option can be used to control how many messages the SNADS channel attempts to send to the remote SNADS side during a particular transaction; if there are more than that number of messages to be processed, the SNADS channel will close the conversation and reopen a new conversation. The default value is 32767.

4.6.4.2.4 Example Option File

A typical `snads_local_option` file might appear as shown in Example 4-3.

Example 4-3 A Sample `snads_local_option` File

```
ADJACENT_NODE_REN=BLUE
BRIDGE_HOST=bridge.example.com
BRIDGE_PORT=9994
BRIDGE_REN=BRIDGE
```

4.6.4.3 Defining the Service

The PMDF-XGS configuration utility creates a `dispatcher_xgs.cnf` file for you, suitable for inclusion into your `dispatcher.cnf` file.

On the machine running PMDF, there are two programs associated with the SNADS Gateway, `snads_master` and `snads_slave`, corresponding to the master (PMDF to SNADS) direction of SNADS channels and the slave (SNADS to PMDF) direction of SNADS channels.

`snads_master` is run automatically by PMDF when it needs to send mail to SNADS.

`snads_slave` must be configured as a service under the PMDF Service Dispatcher. The PMDF Service Dispatcher will then control starting up a `snads_slave` server process to handle requests from the PMDF-XGS transport bridge.

The service definition to be added to the `dispatcher.cnf` file in the PMDF table directory should look, on OpenVMS, something like:

```
[Service=SNADS]
Port=dispatcherport
Image=PMDF_EXE:snads_slave.exe
LogFile=PMDF_LOG:snads_slave.log
```

or, on UNIX, something like:

PMDF-XGS Configuring PMDF-XGS

```
[Service=SNADS]
Port=dispatcherport
Image=/pmdf/bin/snads_slave
LogFile=/pmdf/log/snads_slave.log
```

where *dispatcherport* is the TCP/IP port on the PMDF system to which the PMDF-XGS transport bridge will be making its requests, *i.e.*, that the *recvsrv* pipeline server on the PMDF-XGS transport bridge is configured to use in the *xgs.cmd* file on the PMDF-XGS transport bridge.

See Chapter 11 in the *PMDF System Manager's Guide* for more details on the PMDF Service Dispatcher.

4.6.4.4 An Alias for Sending to the Addressing Channel

If you do not already have an addressing channel definition and corresponding rewrite rules in your PMDF configuration file, see Section 27.1.3 in the *PMDF System Manager's Guide*.

If you already had an addressing channel, make sure you already have (or add now) a rewrite rule for the short form name addressing, *e.g.*,

```
address          $U%youraddressingdomain
```

where *youraddressingdomain* is the domain name you are using for your addressing channel.

Once you have an addressing channel, you will probably want to add an alias that directs messages to the addressing channel for the convenience of your SNADS users. For instance, if you want messages to *MAILMAN@PMDFnode* to be sent to the addressing channel, then add an entry to your PMDF alias file:

```
MAILMAN:         MAILMAN@address
```

(This assumes that you already have an addressing channel with a rewrite rule for the short form name addressing.)

4.6.4.5 Adding MX Records for the SNADS Pseudodomains

If you want other SMTP nodes in your domain or in the outside world to be able to use pseudomain addresses to send mail to users on the SNADS nodes behind the PMDF-XGS gateway, then you will need to add MX records for the SNADS pseudomain names into the DNS.

PMDF-XGS

Detailed Example Configuration

4.7 Detailed Example Configuration

The section shows the details of how the sample site shown in Figure 4–1 would be configured.

Configuration of the SNADS nodes.

1. *Add directory information on the adjacent DISOSS node BLUE for the PMDF OpenVMS system naples.example.com and other systems reached via PMDF. Add this directory information with commands:*

```
ADD      USERTYPE='REMOTE'
         DDN='NAPLES'
         SA='*'
         RGN='          '
         REN='BRIDGE' .
ADD      USERTYPE='REMOTE'
         DDN='MILAN'
         SA='*'
         RGN='          '
         REN='BRIDGE' .
```

2. *Add routing information on the adjacent DISOSS node BLUE for how to send distributions to the OS/2 transport bridge BRIDGE. Add this routing information with the command:*

```
ADD      RGN='          '
         REN='BRIDGE'
         TRANSID='DSVE'
         SSL='*'
         QUEUE=BRID
```

Here BRID is the CICS name for the LU for the BRIDGE system, matching the name used during the CICS configuration step described below in step 3.

3. *Define the SNA link in VTAM and CICS on the adjacent DISOSS node APPLE. The exact specifications required will depend upon what is already defined at a site. For instance, suppose that the control point name of the node BRIDGE, as specified as the SNA node name given in the Communications Manager definition on BRIDGE, is NECTAR. And suppose that CICBLUE is the VTAM application name (APPL) for where DISOSS is running. Then an appropriate VTAM definition might look something like:*

```
LKNECTAR PU  ADDR=04,
             CPNAME=NECTAR
             XID=YES,RESSCB=2
NECTAR LU   LOCADDR=0, DLOGMOD=#BATCH,
             LOGAPPL=CICBLUE
...
CICAPPLE   APPL  ACBNAME(CICBLUE)
```

here LKNECTAR is the PU name given to the node BRIDGE (which need not match any definition on BRIDGE itself). Other parameters can be specified, similar to those for a 3270 connection from BRIDGE to BLUE. A corresponding CICS definition must be created with the commands

PMDF-XGS Detailed Example Configuration

```
CEDA DEFINE GROUP(abc) CONNECTION(BRID)
```

and

```
CEDA DEFINE GROUP(abc) SESSION (xyz)
```

yielding screens

```
CEDA DEFine
BRID
Group
Description ==> PARTNER LU
CONNECTION IDENTIFIERS
Netname      ==> NECTAR
INDsys       ==>
REMOTE ATTRIBUTES
REMOTESystem ==>
REMOTENAME   ==>
CONNECTION PROPERTIES
ACessmethod ==> Vtam  Vtam|IRC|INDirect|Xm
Protocol    ==> Appc  Appc | Lu61
Singlesess  ==> No   Yes | No
DAstream    ==> User  User|3270|SCs|Strf|Lms
RECORDformat ==> U    U | Vb
OPERATIONAL PROPERTIES
+ Autoconnect ==> No   No | Yes | All
INService    ==> Yes  Yes | No
SECURITY
Securityname ==>
Attachsec    ==> Local Local|Identify|Verify
              |Persistent|Hixidpe
BINDPassword ==>      PASSWORD NOT SPECIFIED
BINDSecurity ==> No   No | Yes
```

and

```
CEDA DEFine
Sessions      : xyz
Group         : abc
Description   ==> SESSIONS for PARTNER LU
SESSION IDENTIFIERS
Connection    ==> BRID
SESSName      ==>
NETnameq     ==>
Modename     ==> #BATCH
SESSION PROPERTIES
Protocol      ==> Appc          Appc | Lu61
Maximum      ==> 002 , 000    0-999
RECEIVEPfx   ==>
RECEIVECount ==>              1-999
SENDPfx      ==>
SENDCount    ==>              1-999
SENDSize     ==> 1920         1-30720
RECEIVESize  ==> 1920         1-30720
+ Autoconnect ==> No   No | Yes | All
INService    ==> Yes  Yes | No
```

PMDF-XGS

Detailed Example Configuration

```
SECURITY
Securityname ==>
Attachsec    ==> Local Local|Identify|Verify
              |Persistent|Hixidpe
BINDPassword ==>          PASSWORD NOT SPECIFIED
BINDSecurity ==> No     No | Yes
```

4. Add directory information on additional DISOSS nodes for the PMDF system `naples.example.com` and other systems reached via PMDF. On the node INDIGO, add this directory information with the commands:

```
ADD    USERTYPE='REMOTE'
      DDN='NAPLES'
      SA='*'
      RGN='          '
      REN='BRIDGE' .
ADD    USERTYPE='REMOTE'
      DDN='MILAN'
      SA='*'
      RGN='          '
      REN='BRIDGE' .
```

5. Add routing information on additional DISOSS nodes for how to send distributions to the PMDF-XGS transport bridge BRIDGE. On a DISOSS node such as INDIGO, there should already be a routing table entry describing how to send distributions to BLUE, the SNADS node adjacent to the OS/2 transport bridge. Add a similar entry for the OS/2 transport bridge itself. For instance, if the routing table entry on INDIGO on how to send to BLUE is:

```
ADD    RGN='          '
      REN='BLUE'
      TRANSID='DSVE'
      SSL='*'
      QUEUE='QCOX' .
```

then add a similar routing table entry:

```
ADD    RGN='          '
      REN='BRIDGE'
      TRANSID='DSVE'
      SSL='*'
      QUEUE='QCOX' .
```

6. Add directory information on additional AS/400 nodes. On the nodes AZULE and LAPIS in turn, issue the WRKDIR command for each of NAPLES and MILAN in turn, filling in the resulting screens along the lines of:

PMDF-XGS Detailed Example Configuration

ADD NEW DIRECTORY ENTRY

```

USER
  userid.....: *ANY
  address.....: NAPLES
SYSTEM
  system name.....: BRIDGE
  system group.....: .....
Indirect user.....: N
Print personal mail.....: N

```

ADD NEW DIRECTORY ENTRY

```

USER
  userid.....: *ANY
  address.....: MILAN
SYSTEM
  system name.....: BRIDGE
  system group.....: .....
Indirect user.....: N
Print personal mail.....: N

```

ADD NEW DIRECTORY ENTRY

```

USER
  userid.....: *ANY
  address.....: CUPID
SYSTEM
  system name.....: BRIDGE
  system group.....: .....
Indirect user.....: N
Print personal mail.....: N

```

7. *Add routing information on additional AS/400 nodes for how to send distributions to NAPLES, a.k.a. naples.example.com. Suppose the existing routing table entry on node AZULE for routing to BLUE is:*

ROUTING TABLE ENTRY

```

Destination system
  Name / Group.....: BLUE
  Description.....: node adjacent to XGS bridge
Service level
  Fast:
    Queue name.....: FUJI
    Maximum hops.....: *DFT
  Status:
    Queue name.....: FUJI
    Maximum hops.....: *DFT
  Data high:
    Queue name.....: FUJI
    Maximum hops.....: *DFT
  Data low:
    Queue name.....: FUJI
    Maximum hops.....: *DFT

```

PMDF-XGS

Detailed Example Configuration

Then on AZULE issue the command CFGDSTSVR, select “Routing Table Entry” and create the new routing table entry for routing to BRIDGE:

```
ROUTING TABLE ENTRY
Destination system
  Name / Group.....: BRIDGE
Description.....: PMDF-XGS transport bridge
Service level
  Fast:
    Queue name.....: FUJI
    Maximum hops.....: *DFT
  Status:
    Queue name.....: FUJI
    Maximum hops.....: *DFT
  Data high:
    Queue name.....: FUJI
    Maximum hops.....: *DFT
  Data low:
    Queue name.....: FUJI
    Maximum hops.....: *DFT
```

Suppose that on the node LAPIS, which routes messages to BLUE by way of AZULE, that the existing routing table entry for routing to BLUE is:

```
ROUTING TABLE ENTRY
Destination system
  Name / Group.....: BLUE
Description.....: node adjacent to XGS bridge
Service level
  Fast:
    Queue name.....: YELLOW
    Maximum hops.....: *DFT
  Status:
    Queue name.....: YELLOW
    Maximum hops.....: *DFT
  Data high:
    Queue name.....: YELLOW
    Maximum hops.....: *DFT
  Data low:
    Queue name.....: YELLOW
    Maximum hops.....: *DFT
```

Then on LAPIS, issue the command CFGDSTSVR, select “Routing Table Entry”, and create the new routing table entry for routing to BRIDGE:

```
ROUTING TABLE ENTRY
```

PMDF-XGS Detailed Example Configuration

```

Destination system
  Name / Group.....: BRIDGE
  Description.....: PMDF-XGS transport bridge
Service level
  Fast:
    Queue name.....: YELLOW
    Maximum hops.....: *DFT
  Status:
    Queue name.....: YELLOW
    Maximum hops.....: *DFT
  Data high:
    Queue name.....: YELLOW
    Maximum hops.....: *DFT
  Data low:
    Queue name.....: YELLOW
    Maximum hops.....: *DFT

```

Configuration of the PMDF-XGS transport bridge.

8. *Configure TCP/IP on the PMDF-XGS transport bridge.* We suppose that the site is using OS/2 Warp Connect. If TCP/IP has not already been configured on the OS/2 system, perform the normal TCP/IP configuration using the IBM-supplied "TCP/IP configure" program. Then perform the following additional steps. On the PMDF-XGS transport bridge OS/2 system, edit the file `\mpts\etc\hosts` and if it is not already defined, define the hostname `localhost` as

```
127.0.0.1    localhost
```

Edit the file `\mpts\bin\setup.cmd` and if it is not already present, add the line

```
ifconfig lo 127.0.0.1
```

Use the configuration notebook to configure adding and starting an SNMP daemon supporting the dpi interface; you want the line

```
start /min snmpd -dpi shm -dpi tcp
```

added to the `\mptn\bin\tcpstart.cmd`

9. *Configure the SNA connection on the PMDF-XGS transport bridge.* On the PMDF-XGS transport bridge, run the Communications Manager setup program, `cmsetup.exe`. Assume that the `example.com` site uses `GROLY` as the Network ID for (all) its SNADS nodes. In the local node definition, enter `GROLY` as the Network ID, enter `NECTAR` as the Local Node name, and enter `05D` and `00000` as the Local Node ID. Assume that you are using a LAN connection between the PMDF-XGS transport bridge and the adjacent SNADS node, and that the LAN address of that adjacent SNADS node is `123456789012`, and assume that the VTAM name of the adjacent `BLUE` node is `CPBLUE`. In the SNA connection definition, enter `GROLY` as the Partner Network ID, `CPAPPLE` as the Partner Node Name, and `123456789012` as the LAN Destination Address. In the partner LU definition, enter `GROLY` as the Network ID, `CICBLUE` as the LU Name, and `LOCAL` as the Alias. In the symbolic destination definition, enter `LOCAL` as the Symbolic Destination Name and Partner LU Alias, enter `X'21'002` as the Partner TP, enter `NONE` as the Security, and enter `#BATCH` as the Mode Name.

PMDF-XGS

Detailed Example Configuration

In the transaction program defaults definition, delete the * value from Directory For Inbound Attaches.

Having used Communications Manager for this configuration, a Node Definition File should now be present in Communications Manager's cmlib directory, looking like:

```
DEFINE_LOCAL_CP   FQ_CP_NAME (GROLY.NECTAR      )
                  CP_ALIAS (NECTAR)
                  NAU_ADDRESS (INDEPENDENT_LU)
                  NODE_TYPE (EN)
                  NODE_ID (X'05D00000')
                  NW_FP_SUPPORT (NONE)
                  HOST_FP_SUPPORT (YES)
                  MAX_COMP_LEVEL (NONE)
                  MAX_COMP_TOKENS (0);

DEFINE_LOGICAL_LINK LINK_NAME (LINK0001)
                   FQ_ADJACENT_CP_NAME (GROLY.CBLUE    )
                   ADJACENT_NODE_TYPE (LEARN)
                   DLC_NAME (IBMTRNET)
                   ADAPTER_NUMBER (0)
                   DESTINATION_ADDRESS (X'12345678901204')
                   ETHERNET_FORMAT (NO)
                   CP_CP_SESSION_SUPPORT (NO)
                   SOLICIT_SSCP_SESSION (NO)
                   ACTIVATE_AT_STARTUP (YES)
                   USE_PUNAME_AS_CPNAME (NO)
                   LIMITED_RESOURCE (USE_ADAPTER_DEFINITION)
                   LINK_STATION_ROLE (USE_ADAPTER_DEFINITION)
                   MAX_ACTIVATION_ATTEMPTS (USE_ADAPTER_DEFINITION)
                   EFFECTIVE_CAPACITY (USE_ADAPTER_DEFINITION)
                   COST_PER_CONNECT_TIME (USE_ADAPTER_DEFINITION)
                   COST_PER_BYTE (USE_ADAPTER_DEFINITION)
                   SECURITY (USE_ADAPTER_DEFINITION)
                   PROPAGATION_DELAY (USE_ADAPTER_DEFINITION)
                   USER_DEFINED_1 (USE_ADAPTER_DEFINITION)
                   USER_DEFINED_2 (USE_ADAPTER_DEFINITION)
                   USER_DEFINED_3 (USE_ADAPTER_DEFINITION);

DEFINE_PARTNER_LU  FQ_PARTNER_LU_NAME (GROLY.CICBLUE    )
                  PARTNER_LU_ALIAS (LOCAL)
                  PARTNER_LU_UNINTERPRETED_NAME (CICBLUE  )
                  MAX_MC_LL_SEND_SIZE (32767)
                  CONV_SECURITY_VERIFICATION (NO)
                  PARALLEL_SESSION_SUPPORT (YES);

DEFINE_PARTNER_LU_LOCATION FQ_PARTNER_LU_NAME (GROLY.CICBLUE    )
                          WILDCARD_ENTRY (NO)
                          FQ_OWNING_CP_NAME (GROLY.CPBLUE    )
                          LOCAL_NODE_NN_SERVER (NO);

DEFINE_DEFAULTS  IMPLICIT_INBOUND_PLU_SUPPORT (YES)
                 DEFAULT_MODE_NAME (BLANK)
                 MAX_MC_LL_SEND_SIZE (32767)
                 DEFAULT_TP_OPERATION (NONQUEUED_AM_STARTED)
                 DEFAULT_TP_PROGRAM_TYPE (BACKGROUND)
                 DEFAULT_TP_CONV_SECURITY_RQD (NO)
                 MAX_HELD_ALERTS (10);
```

PMDF-XGS Detailed Example Configuration

```
DEFINE_CPIC_SIDE_INFO SYMBOLIC_DESTINATION_NAME(LOCAL    )
                        PARTNER_LU_ALIAS(LOCAL          )
                        MODE_NAME(#BATCH                )
                        SNA_SERVICE_TP_NAME(X'21',002);

START_ATTACH_MANAGER;
```

10. *Configure the programs on the PMDF-XGS transport bridge.* If you have not yet installed the PMDF-XGS programs on the PMDF transport bridge system, then do so now. The installation procedure will also ask you configuration questions and will generate a `xgs.cmd` in accordance with your answers. Suppose that your CD-ROM drive is `D:` and that you are running the PMDF-XGS transport bridge installation procedure `install.cmd` directly off the PMDF distribution CD-ROM, and that you want to install the PMDF-XGS transport bridge programs in the directory `C:\xgs`.

```
[D:\other\os2\xgs]install.cmd
PMDF / SNADS server installation

Where do you want the code installed [C:\XGS]? C:\XGS
...
PMDF / SNADS server configuration tool
configuring control element

The active components introduce themselves to the control process
using a TCP/IP conversation.
What port is to be used for this conversation [9990]? 9990
The control process needs an SNMP community name to register the
Gateway MIB.
What community name is going to be used [public]? public
One or more SNADS receive processes can be configured
How many receive processes should be run [1]? 1
The receive processes need to connect to the PMDF MTA
What is the TCP/IP hostname of the machine running PMDF? apollo.example.com
What port is the SNADS receive process on the PMDF machine listening on
Port number [9993]? 9993
What is the SNADS name of the Gateway? bridge
What is the maximum document size to be accepted by the Gateway [10000000]? 10000
000
What is the TCP/IP host name of the syslog host [localhost]? localhost
What directory is to be used to store captured SNADS distributions [.\capture]?
.\capture
There can be several SNADS send processes. Each listens on a separate port
and is typically used to send to a different SNADS node.
What port number should the first SNADS send process use [9994]? 9994
What port number should the next SNADS send process use?
(press enter to end)

make sure that c:\xgs is added to the libpath in config.sys

[C:\other\os2\xgs]
```

The resulting `xgs.cmd` file will be:

PMDF-XGS

Detailed Example Configuration

```
start xcontrol public 9990
start recvsrv apollo.example.com 9993 BRIDGE 10000000 localhost 9990 .\capture
start sendsrv 9994 localhost 9990 .\capture
```

Run the `xgs.cmd` procedure.

Shadow the `xgs.cmd` icon to the system's "Startup" folder so that it will be autostarted.

11. *Configure PMDF-XGS on the PMDF system.* Use the PMDF-XGS configuration utility to generate suitable PMDF configuration files.

```
$ PMDF CONFIGURE XGS
```

```
PMDF-XGS configuration utility, Version 5.1
```

```
...
```

```
SNADS name of this PMDF system known to the SNADS world []? APOLLO
```

```
TCP/IP name of the OS/2 system acting as the bridge []? bridge.example.com
```

```
SNADS REN name of the OS/2 system acting as the bridge []? BRIDGE
```

```
TCP/IP port number the sendsrv on the OS/2 system listens on [9994]? 9994
```

```
TCP/IP port number the snads_slave program listens on [9993]? 9993
```

```
SNADS REN name of the SNADS system immediately adjacent to the OS/2 bridge []?
B LUE
```

```
Domain name corresponding to previous SNADS system []? blue.example.com.
```

```
SNADS RGN name of the SNADS system immediately adjacent to the OS/2 bridge [none]?
```

```
Name of a SNADS system to which PMDF will route mail [RETURN if no more]? INDIGO
```

```
Domain name corresponding to previous SNADS system []? indigo.example.com
```

```
Name of a SNADS system to which PMDF will route mail [RETURN if no more]? AZULE
```

```
Domain name corresponding to previous SNADS system []? azule.example.com
```

```
Name of a SNADS system to which PMDF will route mail [RETURN if no more]? LAPIS
```

```
Domain name corresponding to previous SNADS system []? lapis.example.com
```

```
Name of a SNADS system to which PMDF will route mail [RETURN if no more]? 
```

```
Domain name of an SMTP system from which PMDF will route mail []? milan.example.com
```

```
Domain name of an SMTP system from which PMDF will route mail []? 
```

```
...
```

The resulting `xgs.chans` file will have a channel definition

```
snads_local defragment 733 header_733 maxheaderaddrs 1 \
  addrsperfile 256 charset7 US-ASCII charset8 ISO-8859-1
SNADS-DAEMON NAPLES
```

and the `xgs.rules` file will have rewrite rules

PMDF-XGS Detailed Example Configuration

```
INDIGO                                $U%indigo.example.com
                                      $U%INDIGO@SNADS-DAEMON$E$F
indigo.example.com                    $U%INDIGO@SNADS-DAEMON$Qsnads_local
indigo.example.com                    $U%indigo@SNADS-DAEMON
AZULE                                  $U%AZULE.example.com
azule.example.com                     $U%AZULE@SNADS-DAEMON$E$F
azule.example.com                     $U%AZULE@SNADS-DAEMON$Qsnads_local
azule.example.com                     $U%banana.example.com@SNADS-DAEMON
LAPIS                                  $U%lapis.example.com
lapis.example.com                     $U%LAPIS@SNADS-DAEMON$E$F
lapis.example.com                     $U%LAPIS@SNADS-DAEMON$Qsnads_local
lapis.example.com                     $U%lapis.example.com@SNADS-DAEMON
BLUE                                   $U%lapis.example.com
blue.example.com                       $U%BLUE@SNADS-DAEMON$E$F
blue.example.com                       $U%BLUE@SNADS-DAEMON$Qsnads_local
blue.example.com                       $U%blue.example.com@SNADS-DAEMON
milan                                  $U%milan.example.com$Msnads_local
```

and the `snads_local_option.` file will be

```
ADJACENT_NODE_REN=BLUE
BRIDGE_HOST=bridge.example.com
BRIDGE_PORT=9994
BRIDGE_REN=BRIDGE
```

and the `dispatcher_xgs.cnf` file will be

```
[service=SNADS]
port=9993
image=PMDF_EXE:SNADS_SLAVE.EXE
logfile=PMDF_LOG:SNADS_SLAVE.LOG
```

As the `xgs.checklist` directs, edit your `pmdf.cnf` file to insert or activate references to `xgs.chans` and `xgs.rules` and edit your `dispatcher.cnf` file to activate the reference to `dispatcher_xgs.cnf` and edit your PMDF alias file to insert the alias to be directed to the addressing channel.

12. If you are using a compiled configuration, recompile and reinstall it. Restart the PMDF Service Dispatcher.
13. Try sending a test message from PMDF to SNADS, and another test message from SNADS to PMDF.

4.8 Multiple Connections Using Multiple `snads_` Channels

It is possible to define links to several SNADS nodes from PMDF. This can improve the efficiency of the SNADS side of your mail network. It can also be suitable if your SNADS network is not particularly hierarchical or if there is no natural “main hub” SNADS node to place “adjacent” to the PMDF-XGS transport bridge; in such a case it can be more natural to have a number of nodes “directly adjacent” to the PMDF-XGS transport bridge, and to let those nodes have their own direct connection to the PMDF-XGS transport bridge and through it, to the PMDF system itself.

PMDF-XGS

Multiple Connections Using Multiple snads_ Channels

Each SNADS link will need a separate PMDF channel associated with it, `snads_snadsnodename`. The `snadsnodename` part of the channel name is quite significant here; this (lower case) string will be converted to upper case and used as the CPI-C Symbolic Destination Name to reach the specified SNADS node. For mail arriving from SNADS nodes, PMDF will look at the partner LU alias for the node from which the message is arriving and process the message with the correspondingly named channel. (Indeed, this is true for the `snads_local` channel as well, which is why the Symbolic Destination Name for that channel must be LOCAL.) The `snads_local` channel is used for messages arriving from SNADS nodes which haven't been defined, or from nodes whose alias does not start with an alphabetic character.

4.8.1 Configuring the Additional Adjacent SNADS Nodes

Each additional SNADS node which is to have its own channel to PMDF must have its routing information updated so that it routes distributions directly to the OS/2 transport bridge. For each additional DISOSS node, see Section 4.6.1.2; for each additional AS/400 node, see Section 4.6.2.2.

4.8.2 Adding Additional Definitions and Servers on the PMDF-XGS Transport Bridge

To use multiple channels, the PMDF-XGS transport bridge Communications Manager configuration and `xgs.cmd` must be modified.

4.8.2.1 Adding Additional Definitions on the PMDF-XGS Transport Bridge

You must edit the Communications Manager configuration to define the additional partner LU's as symbolic destinations.

4.8.2.2 Adding Additional Servers on the PMDF-XGS Transport Bridge

The startup procedure on the PMDF-XGS transport bridge will need to start up additional send processes on additional ports; *e.g.*, the `xgs.cmd` file should have the format shown in Figure 4–15 on NT, or the format shown in Figure 4–16 on OS/2.

Figure 4–15 Format of the `xgs.cmd` File for Multiple Send Processes on NT

```
start sendsrv bridgeport1 [loghost1]
start sendsrv bridgeport2 [loghost2]
start sendsrv bridgeport3 [loghost3]
...
start recvsrv PMDFhost dispatcherport bridgeREN size [loghost]
```

PMDF-XGS

Multiple Connections Using Multiple `snads_` Channels

Figure 4–16 Format of the `xgs.cmd` File for Multiple Send Processes on OS/2

```
start xcontrol
start sendsrv bridgeport1 [loghost1] [SNMPport] [senddirectory1]
start sendsrv bridgeport2 [loghost2] [SNMPport] [senddirectory2]
start sendsrv bridgeport3 [loghost3] [SNMPport] [senddirectory3]
...
start recvsrv PMDFhost dispatcherport bridgeREN size [loghost] [SNMPport] [recvdirectory]
```

where `sendport1`, `sendport2`, `sendport3`, *etc.*, are distinct port numbers corresponding to the port numbers that the `snads_channels`' option files specify, where `senddirectory1`, `senddirectory2`, `senddirectory3`, *etc.*, are a capture directory or directories on the PMDF-XGS transport bridge to which copies of messages from PMDF to SNADS will be written, where `PMDFhost` is the TCP/IP name of the PMDF system where `dispatcherport` is the port that the `snads_slave` server on the PMDF system is configured to listen on in the `dispatcher.cnf` file, where `bridgeREN` is the TCP/IP name of the PMDF-XGS transport bridge, and where `size` is the maximum size of SNADS distributions accepted, and where `recvdirectory` is the capture directory to which copies of messages from SNADS to PMDF are written.

Note that no additional `xcontrol` or `recvsrv` processes are needed. However, for performance reasons, you can want to enable several additional `recvsrv` servers to enable the PMDF-XGS transport bridge to have multiple connections open to the PMDF system at the same time, thereby allowing more throughput. Note that such additional `recvsrv` servers would all use the same PMDF Service Dispatcher port number.

These new SNADS send servers on the PMDF-XGS transport bridge know the proper SNADS destination from the name of the `snads_snadsnodename` channel which connects to their port; they use the `snadsnodename` of the channel which connects to their port as the Symbolic Destination Name.

4.8.3 Adding Additional Channels to the PMDF Configuration

You will need to add additional channels and corresponding rewrite rules to your PMDF configuration file, and create corresponding additional option files.

4.8.3.1 Adding the Channel Definitions and Rewrite Rules

The additional channel definitions should be along the lines of:

```
snads_snadsnode1 defragment 733 header_733 maxheaderaddrs 1 addrspersfile 256 \
  charset7 US-ASCII charset8 ISO-8859-1
snadsnode1.yourdomain SNADS-name-for-PMDF-system

snads_snadsnode2 defragment 733 header_733 maxheaderaddrs 1 addrspersfile 256 \
  charset7 US-ASCII charset8 ISO-8859-1
snadsnode2.yourdomain SNADS-name-for-PMDF-system

...
```

where `snadsnode1.yourdomain` is the pseudo domain name associated with `snadsnode1`, *i.e.*, the name by which that SNADS node is known from the PMDF side,

PMDF-XGS

Multiple Connections Using Multiple snads_ Channels

and where *SNADS-name-for-PMDF-system* is the name by which the PMDF system is known from the SNADS side. And the additional channels should have corresponding rewrite rules:

```
snadsnode1          $U%snadsnode1.yourdomain
snadsnode1.yourdomain $U@snadsnode1.yourdomain
snadsnode2          $U%snadsnode2.yourdomain
snadsnode2.yourdomain $U@snadsnode2.yourdomain
```

and to handle the SNADS side's short form names for the PMDF system and any other explicitly addressable systems, additional rewrite rules:

```
PMDFnode           $U%PMDFdomainname$Msnads_snadsnode1
othernodeonPMDFside1 $U%otherdomain1$Msnads_snadsnode1
othernodeonPMDFside2 $U%otherdomain2$Msnads_snadsnode1
...
PMDFnode           $U%PMDFdomainname$Msnads_snadsnode2
othernodeonPMDFside1 $U%otherdomain1$Msnads_snadsnode2
othernodeonPMDFside2 $U%otherdomain2$Msnads_snadsnode2
...
```

4.8.3.2 Additional Channel Option Files

Each new channel must have its own corresponding option file. See Section 4.6.4.2.

4.8.4 Example Configuration with Multiple snads_ Channels

If the connection to BLUE is channel `snads_local`, the connections to AZULE and INDIGO might be defined as channels `snads_azule` and `snads_indigo`. The important point to remember about the <SNADS\smallcaps> channel names is that for a channel `snads_snadsnodename`, the `snadsnodename` is converted to uppercase, and is used as the CPI-C symbolic destination name to reach that node. Mail arriving from the SNADS nodes is associated with the appropriate channel by the partner LU alias from which the mail arrived. This means that the partner LU alias for node AZULE must be `azule` because the channel name is `snads_azule`. This was just as important for the `snads_local` channel, which is why the symbolic destination had to be `LOCAL`, but for mail arriving from SNADS, the `snads_local` channel is also used for mail coming from nodes which haven't been defined. Any mail arriving from a node which have an alias which does not start with an alphabetic character will be processed through the `snads_local` channel.

5 The MultiWare Client API Approach in PMDF-LAN (OpenVMS)

The MultiWare client API in our MultiNet TCP/IP package allowed an OpenVMS system direct access to a NetWare file server. MultiWare is no longer supported by Process Software.

The MultiWare client API support code in PMDF is still present, and should still work as well as it ever has, for sites that have been using it and want to continue doing so. But if there are problems using it, they will not be able to be resolved. New sites should use a different method, and old sites using the MultiWare client API method should plan on migrating to another method.

The sections below describe use of this technique.

5.1 Direct OpenVMS Client Access to a NetWare File Server

Use of the MultiWare client API allows an OpenVMS process to read and write files directly from and to a Novell NetWare file server's disks. It only applies when all four of the following conditions are met:

1. The OpenVMS system hosting PMDF has MultiNet V3.2D² or later installed which includes the MultiWare client API library.
2. You have configured MultiNet to turn on the IPX/SPX network protocol stack for your network device. See the MultiNet manuals for information now how to enable the LAN interface for IPX. You need to provide the NetWare network number and encapsulation type being used on your existing NetWare network.
3. The PC mail system is stored on a NetWare file server.
4. The PC mail system's NetWare file server and the OpenVMS system are on the same physical, usually Ethernet, network or on the same logical network by having Novell packets bridged or routed. A logical network such as this is sometimes referred to as an extended LAN.

When this is your configuration, there is no need for the transfer PC. PMDF uses the MultiWare client API library, included with MultiNet, to retrieve inbound messages directly from the NetWare server. PMDF also creates outbound messages directly on that same server's disks. All transfers are completely controlled and managed by processes on the OpenVMS system.

² MultiNet V3.2B and C can not work in your network configuration. They are identical in that they only support Ethernet (Novell Ethernet_II) and 802.2 (Novell Ethernet_SNAP) encapsulations. Also, revisions B and C do not correctly handle dropped packets. Revision D supports all encapsulations.

The MultiWare Client API Approach in PMDF-LAN (OpenVMS)

Configuring Use of the MultiWare Client API Approach in PMDF-LAN Channels

5.2 Configuring Use of the MultiWare Client API Approach in PMDF-LAN Channels

PMDF-LAN channel option files have an `ACCESS_METHOD` option that specifies the access method PMDF-LAN will use to read and write message files. On OpenVMS, a value of 1 selects direct NetWare file server access using the MultiWare client API library. When `ACCESS_METHOD=1` is specified, the `FILE_SERVER`, `USERNAME`, and `PASSWORD` options must also be specified. `ACCESS_METHOD=1` is not supported on UNIX.

Note that when `ACCESS_METHOD=1` is set, you will likely want to protect your channel option file against general access as it will necessarily contain a username and password for one of your NetWare file servers.

FILE_SERVER (string)

Specifies the name of the NetWare file server to use when `ACCESS_METHOD=1` is specified.

PASSWORD (string)

Specifies the password that goes with the NetWare username specified with the `USERNAME` option. This option is used only when `ACCESS_METHOD=1` is specified.

USERNAME (string)

Specifies the NetWare username to use when the PMDF process logs into a file server. This option is used only when `ACCESS_METHOD=1` is specified.

For instance, an example of a `cc:Mail` channel option file when when PMDF directly accesses messages stored on a NetWare file server by using MultiNet's MultiWare client API library is shown in Example 5-1; an example for a Microsoft Mail channel option file is shown in Example 5-2; an example for an MHS channel option file is shown in Example 5-3; an example for a GroupWise (WordPerfect Office) channel option file is shown in Example 5-4.

Remember that the MS-DOS directory and file specifications are restricted to an eight character name part and three character extension part.

Example 5-1 A Sample cc:Mail Channel Option File

```
ACCESS_METHOD=1
FILE_SERVER=DOGHUT
USERNAME=MROCHEK
PASSWORD=WOOF
!
CC_GATEWAY_NAME=PMDF
CC_OUTPUT_FILE_NAME=SYS:/MROCHEK/CCMAIL.IMP
CC_INPUT_FILE_NAME=SYS:/MROCHEK/CCMAIL.EXP
CC_UNDEL_FILE_NAME=SYS:/MROCHEK/CCMAIL.UND
```

The MultiWare Client API Approach in PMDF-LAN (OpenVMS)

Configuring Use of the MultiWare Client API Approach in PMDF-LAN Channels

Example 5-2 A Sample Microsoft Mail Channel Option File

```
ACCESS_METHOD=1
FILE_SERVER=FRODO
USERNAME=JDOE
PASSWORD=MORGAN
!
FF_DEFAULT_NETWORK=EXAMPLE
FF_DEFAULT_POSTOFFICE=HQ
!
FF_OUTPUT_FILE_NAME=SYS:/MSMAIL/EXAMPLE/PMDF2MS.MSG
FF_INPUT_FILE_NAME=SYS:/MSMAIL/EXAMPLE/MS2PMDF.MSG
```

Example 5-3 Sample MHS Channel Option File

```
ACCESS_METHOD=1
FILE_SERVER=FRODO
USERNAME=JDOE
PASSWORD=MORGAN
!
MHS_GATEWAY_WORKGROUP=PMDF
MHS_GATEWAY_USERNAME=MAILER
MHS_GATEWAY_TYPE=SMTP
MHS_ID_SUFFIX=81BADD28
!
MHS_DEFAULT_WORKGROUP=EXAMPLE
!
MHS_IN=SYS:/MHS/MAIL/GATES/PMDF/OUT/
MHS_IPARCEL=SYS:/MHS/MAIL/GATES/PMDF/OPARCEL/
MHS_OPARCEL=SYS:/MHS/MAIL/GATES/PMDF/IPARCEL/
MHS_OUT=SYS:/MHS/MAIL/GATES/PMDF/IN/
```

Example 5-4 A Sample GroupWise Channel Option File

```
! Sample PMDF_TABLE:wpo_local_option. file on OpenVMS
ACCESS_METHOD=1
FILE_SERVER=DOGHUT
USERNAME=MROCHEK
PASSWORD=WOOF
!
WPO_GATEWAY_DOMAIN=PROCESS
WPO_GATEWAY_PO=PMDF
!
WPO_DEFAULT_DOMAIN=EXAMPLE
WPO_DEFAULT_PO=HQ
!
WPO_API_IN=SYS:/WPO/EXAMPLE/WPGATE/APIGATE/API_OUT/
WPO_ATT_IN=SYS:/WPO/EXAMPLE/WPGATE/APIGATE/ATT_OUT/
WPO_API_OUT=SYS:/WPO/EXAMPLE/WPGATE/APIGATE/API_IN/
WPO_ATT_OUT=SYS:/WPO/EXAMPLE/WPGATE/APIGATE/ATT_IN/
```

6 MultiNet SNMP Subagent for PMDF (OpenVMS)

The former MultiNet SNMP subagent which was used to serve out PMDF channel counters is no longer available, and not supported by Process Software.

7 MultiNet MM User Agent (OpenVMS)

As there is no longer a vendor supporting the former MultiNet MM user agent, Process Software can no longer continue to support the MultiNet MM interface in PMDF. The actual code in PMDF for supporting MultiNet MM has not yet been removed from PMDF and it should continue to work as well as it ever did; but problems discovered in it cannot be resolved. New sites should not use this and old sites using MultiNet MM should plan on migrating.

MultiNet MM supports PMDF as one of several possible message transfer agents. The MM user should not be aware of any difference between PMDF and any other transfer agent.

7.1 Sending Messages with MultiNet MM

MultiNet used to include support for PMDF in its MM user agent. MultiNet MM will automatically detect and use PMDF if it is available. No configuration changes are needed to enable this feature.

7.2 Receiving Messages with MultiNet MM

PMDF provides support for delivery to MultiNet MM's local mailboxes. By default, such delivery is disabled (unless the user has profiled Multinet MM as their preferred mailbox format). To enable it based merely upon the presence of certain files, described below, specify `MULTINET_MM_EXCLUSIVE=0` in the PMDF option file. (In addition, the PMDF option `VMS_MAIL_EXCLUSIVE`, 0 by default, must not be positive.) See the *PMDF System Manager's Guide* for a description of these options.

When delivering local mail, PMDF checks each address to see if it corresponds to a local user. If it does, then depending upon the setting of the PMDF options `MULTINET_MM_EXCLUSIVE` and `VMS_MAIL_EXCLUSIVE`, PMDF may then check to see whether that user has MM mailbox files in his or her home directory. The names of these files are `mail.txt` and `$hdrs$mail.txt`; the former contains message text while the latter contains pointers to individual message headers in `mail.txt`. If these files exist and if the above mentioned PMDF options are appropriately set, then PMDF appends the new message to the message file and updates the header pointer file. In this case delivery via VMS MAIL is entirely circumvented.

If any of the conditions listed are not met, PMDF delivers the message using VMS MAIL. This delivery strategy is not enabled by default; configuration changes must be made to enable it.

Delivery to MM can be disabled unconditionally by setting `MULTINET_MM_EXCLUSIVE=-3` or by setting `VMS_MAIL_EXCLUSIVE=3` in the PMDF option file.

MultiNet MM User Agent (OpenVMS) Receiving Messages with MultiNet MM

Delivery to VMS MAIL can be disabled unconditionally by setting `VMS_MAIL_EXCLUSIVE=-3` or by setting `MULTINET_MM_EXCLUSIVE=3` in the PMDF option file.

7.3 The PMDF Option File and MultiNet MM

The following MultiNet MM-related options are available in the PMDF option file:

MULTINET_MM_EXCLUSIVE (-3 to 3; OpenVMS only)

The `MULTINET_MM_EXCLUSIVE` option controls whether or not PMDF's local channel delivers exclusively to MultiNet MM or not. Possible values are:

Value	Usage
-3	Never deliver mail to MultiNet MM mailboxes.
-2	Not used at present; do not specify.
-1	Deliver mail to MultiNet MM mailboxes only if MultiNet MM is specified as part of the user's profile information.
0	Deliver mail to MultiNet MM mailboxes if the user has the appropriate MultiNet MM mailbox files in his or her home directory.
+1	Deliver mail to the user's Multinet MM mailbox by default unless the user's profile information specifies some other type of delivery.
+2	Not used at present; do not specify.
+3	Deliver mail to the user's Multinet MM mailbox unconditionally.

The default is `-1`, which means that MultiNet MM delivery will be used only when the user has profiled MultiNet MM as his or her preferred mailbox format.

VMS_MAIL_EXCLUSIVE (-3 to 3; OpenVMS only)

The `VMS_MAIL_EXCLUSIVE` option controls whether or not PMDF's local channel delivers exclusively to VMS MAIL or not. Possible values are:

Value	Usage
-3	Never deliver mail to VMS MAIL mailboxes.
-2	Not used at present; do not specify.
-1	Deliver mail to VMS MAIL mailboxes only if VMS MAIL is specified as part of the user's profile information.
0	Deliver mail to VMS MAIL mailboxes unless the user has some other sort of mailbox in his or her home directory or has specified some other kind of delivery as part of his or her profile information.
+1	Deliver mail to the user's VMS MAIL mailbox by default unless the user's profile information specifies some other type of delivery.
+2	Not used at present; do not specify.
+3	Deliver mail to the user's VMS MAIL mailbox unconditionally.

The default is `0`, which means that either VMS MAIL is used unless a given user has the necessary MultiNet MM files in his or her home directory or has profiled the use of MultiNet MM.

8 Utilities

This chapter describes the obsolete PMDF PUNCH utility.

Utilities

PUNCH

PUNCH— Convert PUNCH files

Convert files in LISTSERV PUNCH format to regular text files.

restrictions *None.*

SYNTAX **PMDF PUNCH** *PUNCH-file-spec [output-file-spec]*

Command Qualifiers	Defaults
<i>None.</i>	<i>None.</i>

prompts PUNCH input: *PUNCH-file-spec*
Output file: *output-file-spec*

PARAMETERS

PUNCH-file-spec

The name of a LISTSERV PUNCH file. Only a single file name can be specified; wildcards can not be used.

output-file-spec

The name of the output file is determined from the PUNCH /ID line in the input file. Specifying the output file name on the command line will override the PUNCH /ID line default.

DESCRIPTION

PUNCH is a utility which converts files stored in BITNET LISTSERV PUNCH format into regular text files. LISTSERV PUNCH format is used to store files containing records longer than 80 characters in a format that can be transmitted across BITNET (which imposes a record length limit of 80 characters on many file types).

PUNCH was originally written by Eric Thomas and adapted for use in PMDF by Ned Freed.

Index

\$hdrs\$mail.txt file
See MultiNet MM

A

Addresses

See also SNADS channels
SNADS • 4-3

Addressing channels

use with SNADS channels • 4-33

ANJE channels

See BITNET channels

availability of PMDF • vii

B

BITNET channels • 3-1 to 3-23

8 x 8 address restrictions • 3-17

accessible systems • 3-6

bitnet_domains_driver.com

See bitnet_domains_driver.com file

BN_SLAVE • 3-14 to 3-16

PMDF_DISABLE_BN_SLAVE logical • 3-15

restarting • 3-15

starting • 3-15

stopping • 3-15

temporarily stopping • 3-15

broadcast messages • 2-2, 3-14

buildmfdisp.com utility • 3-18

configuration

ANJE • 3-3 to 3-4

Jnet • 3-2 to 3-3

envelope To: addresses • 3-16 to 3-17

handling long usernames • 3-17

lmd.com file • 3-14 to 3-16

local mail delivery demon • 3-14 to 3-16

local mailer gateways • 3-20 to 3-23

local NJE mailers • 3-20 to 3-23

mailer access • 3-6 to 3-7

maintenance • 3-13

mfdisp.exe image • 3-18

multigate keyword • 3-12 to 3-13

node entry changes • 3-12

nolocaluser keyword • 3-14

option file

base notation • 3-5

format • 3-5

location • 3-5

BITNET channels (cont'd)

options • 3-4 to 3-5

CLASS • 3-5

EXPAND_TABS • 3-5

FORMAT • 3-5

LINE_LENGTH • 3-5

OPTIONS • 3-5

RECORDSIZE • 3-5

PMDF_DISABLE_BN_SLAVE logical • 3-15

progress messages • 2-2, 3-14

registering a mailer • 3-21 to 3-23

SENDFILE-F-MYNODE Jnet error • 3-7

servers tag • 3-21 to 3-23

systems not running NJE • 3-18 to 3-19

TICK BSMTP command • 3-12

trusted mailer • 3-12

updating routing information • 3-13

VERB BSMTP command • 3-12

version compatability

ANJE • 3-1

Jnet • 3-1

BITNET PUNCH files • 8-2

bitnet_domains_driver.com file • 3-7 to 3-11

options

BITNET_ALIAS • 3-10

BITNET_CHANNEL_TYPE • 3-10

BITNET_LOCAL_CHANNEL • 3-10

BITNET_QUEUE • 3-10

CHANNEL_KEYWORD_OPTIONS • 3-11

DO_CRDB_BR • 3-10

DO_CRDB_GR • 3-10

FORCE_BITNET • 3-9

FORCE_INTERBIT • 3-10

FORCE_INTERNET • 3-9

GATEWAY_SYSTEM_NAME • 3-11

GATEWAY_USER_NAME • 3-11

HANDLES_MX • 3-8

INTERNET_CHANNELS • 3-9

INTERNET_GATEWAY • 3-9

INTERNET_GATEWAY_HOST • 3-9

ON_INTERNET • 3-8

PERIOD • 3-10

SKIP_DOMAIN • 3-9

SKIP_XMAILER • 3-9

TCP_DAEMON • 3-9

BN_SLAVE

See BITNET channels

BN_SLAVE daemon

failure of

log file • 3-15

Index

C

Carosso, John • 3-2
cc:Mail channels
 options
 FILE_SERVER • 5-2
 PASSWORD • 5-2
 USERNAME • 5-2
 transferring messages
 NetWare file server • 5-1
channels
 ANJE
 See BITNET channels
 BITNET
 See BITNET channels
 Jnet
 See BITNET channels
Channels
 SNADS
 See SNADS channels
comment lines
 in BITNET channel option files • 3-5
Communications Manager/2
 on PMDF-XGS transport bridge • 4-2
CRDB utility
 use with BITNET channels • 3-10
Crosswell, Alan • 3-2, 3-12

D

daemon keyword • 2-1 to 2-2, 3-6
Dispatcher
 SNADS channels • 4-32
DOMAIN NAMES file
 obtaining • 3-7
DOMAINS NAMES file
 usage • 3-7

E

errors
 SENDFILE-F-MYNODE Jnet error • 3-7
 unknown BITNET hosts • 3-13
exclamation point
 comment indicator
 BITNET channel option files • 3-5

F

files
 \$hdrs\$mail.txt
 See Multinet MM
 bgateway.rules • 3-8
 bitnet.rules • 3-8
 bitnet_domains_driver.com
 See bitnet_domains_driver.com file
 bn_slave.log • 3-15
 buildmfdisp.com • 3-18
 domain.dat • 3-8
 domains.names
 See DOMAINS NAMES file
 gates.chans • 3-8
 gates.rules • 3-8
 gateway.rules • 3-8
 link_username.com • 3-1, 3-4
 link_username.exe • 3-4
 lmd.com
 See BITNET channels
 PMDF version vs. Jnet version • 3-14
 mail.txt
 See MultiNet MM
 mfdisp.exe • 3-18
 tcpip.rules • 3-8
 username.exe • 3-4
 xmailer.names
 See XMAILER NAMES file
Files
 dispatcher_xgs.cnf • 4-32
 PUNCH • 8-2

H

\$hdrs\$mail.txt file
 See MultiNet MM

I

IBM mainframe mail systems
 See SNADS channels
INTERBIT gateway • 3-6

J

Jnet channels
See BITNET channels

K

keywords
 contchar • 2-2
 contposition • 2-2
 daemon • 2-1 to 2-2, 3-6
 localuser • 2-2
 multigate • 2-1 to 2-2, 3-12 to 3-13
 nocaluser • 2-2, 3-14
 nomultigate • 2-1 to 2-2
 notick • 2-3, 3-12
 tick • 2-3, 3-12
 user • 2-3, 3-6, 3-12
 verb_never • 2-4
 verb_none • 2-4
 verb_off • 2-4, 3-12
 verb_on • 2-4, 3-12

L

LISTSERV PUNCH files • 8-2
 lmd.com file
 See BITNET channels
 local mail delivery demon
 See BITNET channels
 localuser keyword • 2-2
 logical names
 JAN_BN_MASTER_FLAGS • 3-7
 JAN_SENDFILE_FLAGS • 3-7
 PMDF_CHANNEL • 3-14 to 3-15, 3-16
 PMDF_DISABLE_BN_SLAVE • 3-15

M

mail
 MultiNet MM
 receiving • 7-1
 sending • 7-1
 mail.txt file
 See MultiNet MM

Miller, Ed • 3-2
 MM
 See MultiNet MM
 monitoring
 SNMP
 MultiNet subagent
 availability • 6-1
 MS mail channels
 transferring messages
 NetWare file server • 5-1
 transferring messages
 NetWare file server • 5-1
 multigate keyword • 2-1 to 2-2, 3-12 to 3-13
 MultiNet MM
 \$hdrs\$mail.txt file • 7-1
 disabling • 7-1
 exclusive use • 7-2
 mail.txt file • 7-1
 MULTINET_MM_EXCLUSIVE option • 7-2
 receiving mail to • 7-1
 sending mail with • 7-1

N

nocaluser keyword • 2-2, 3-14
 nomultigate keyword • 2-1 to 2-2
 notick keyword • 2-3, 3-12
 Novell MHS channels
 transferring messages
 NetWare file server • 5-1

O

Ordering PMDF • vii
 OS/2 TCP/IP
 on PMDF-XGS transport bridge • 4-2
 OS/2 Warp Connect
 on PMDF-XGS transport bridge • 4-2

P

PC channels
 transferring messages
 NetWare file server • 5-1
 percent sign
 base notation
 in BITNET channel option files • 3-5

Index

PMDf options
 MULTINET_MM_EXCLUSIVE • 7-2
 VMS_MAIL_EXCLUSIVE • 7-1, 7-2
Process Software, LLC • vii
PUNCH utility • 8-2

R

RESTART utility
 BN_SLAVE • 3-13, 3-15

S

SHUTDOWN utility
 BN_SLAVE • 3-15
SMTP commands
 example • 3-21
SNADS channels • 4-1 to 4-46
 addresses • 4-6
 AS/400 node configuration • 4-14 to 4-19
 attachments • 4-5
 configuration • 4-29 to 4-33
 DISOSS node configuration • 4-9 to 4-14
 dispatcher service configuration • 4-32
 message structure • 4-5
 notification messages • 4-5
 option file
 format • 4-31
 location • 4-30
 options • 4-31 to 4-32
 ADJACENT_NODE_REN • 4-31
 ADJACENT_NODE_RGN • 4-31
 BRIDGE_HOST • 4-31
 BRIDGE_PORT • 4-31
 BRIDGE_REN • 4-31
 SAVE_HEADERS • 4-31
 SUPPRESS_DELIVERY_RECEIPTS • 4-31
 SUPPRESS_READ_RECEIPTS • 4-31
 snads_master program • 4-32
 snads_slave program • 4-32
 status distributions • 4-5
 transport bridge
 configuration • 4-19 to 4-29
 function of • 4-4
 required software • 4-2

T

Thomas, Eric • 8-2
tick keyword • 2-3, 3-12
Transport bridge
 PMDf-XGS
 function of • 4-4
 required software • 4-2
trusted mailers
 See BITNET channels

U

user keyword • 2-3, 3-6, 3-12
username.exe image • 3-4
utilities
 CONFIGURE • 1-1
 CRDB
 use with BITNET channels • 3-10
 RESTART
 BN_SLAVE • 3-13, 3-15
 SHUTDOWN
 BN_SLAVE • 3-15
Utilities • 8-1 to Index-1
 PUNCH • 8-2

V

verb_never keyword • 2-4
verb_none keyword • 2-4
verb_off keyword • 2-4, 3-12
verb_on keyword • 2-4, 3-12

W

WPO channels
 transferring messages
 NetWare file server • 5-1

X

XMAILER NAMES file
 obtaining • 3-7
 usage • 3-7