IBM System z Personal Development Tool: Volume 2
Installation and Basic Use

June 2013
Note: Before using this information and the product it supports, read the information in “Notices” on page v.

Sixth Edition (June 2013)

This edition applies to the IBM 1090 system (known as zPDT) that is available at the time of publication, which is Version 1 Release 4 with fixpack 1.

© Copyright International Business Machines Corporation 2009, 2013. All rights reserved.
Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
Contents

Notice ................................................................. v
Trademarks .......................................................... vi

Preface ................................................................. vii
The author .......................................................... vii
Now you can become a published author, too! .......................... vii
Comments welcome ................................................. viii
Stay connected to IBM Redbooks .................................... viii

Chapter 1. Linux and zPDT installation ........................................ 1
1.1 Disk planning ..................................................... 3
1.2 Linux installation .................................................. 4
   1.2.1 x3270 keyboard maps .......................................... 5
   1.2.2 Other Linux notes .............................................. 6
1.3 Install zPDT software ............................................. 7
   1.3.1 Alter Linux files ............................................... 9
1.4 zPDT serial numbers ............................................. 11
1.5 Token activation and renewal .................................... 11
1.6 Installing a new zPDT release .................................... 13

Chapter 2. AD CD z/OS installation .......................................... 15
2.1 General principles ................................................ 16
2.2 System z operating systems ...................................... 16
   2.2.1 Media ......................................................... 16
2.3 Packages for zPDT ................................................. 17
2.4 Installing an AD-CD system ...................................... 17
   2.4.1 Specific installation instructions ........................... 17
   2.4.2 IODF device numbers ....................................... 18
   2.4.3 zPDT control files .......................................... 19
   2.4.4 IPL and operation ........................................... 20
   2.4.5 Shutting down ............................................... 21
   2.4.6 Startup messages .......................................... 22
   2.4.7 Local volumes .............................................. 22
2.5 z/OS parameters ................................................ 23
2.6 Multiple operating systems ..................................... 23

Chapter 3. LANs and TCP/IP .............................................. 25
3.1 Overview of LAN usage .......................................... 26
   3.1.1 Three 3270 interfaces ....................................... 26
3.2 Basic QDIO setup ............................................... 27
3.3 Five scenarios ................................................... 28
   3.3.1 Scenario 1 .................................................... 30
   3.3.2 Scenario 2 .................................................... 31
   3.3.3 Scenario 3 .................................................... 33
   3.3.4 Scenario 4 .................................................... 35
   3.3.5 Scenario 5 .................................................... 37
   3.3.6 z/OS resolver ............................................... 39
   3.3.7 Debugging LAN setups .................................... 41
   3.3.8 Performance problems .................................... 42
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.
Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

- CICS®
- DB2®
- IBM®
- MVS™
- Parallel Sysplex®
- PartnerWorld®
- Redbooks®
- Redbooks (logo) ®
- Resource Link®
- S/390®
- System z®
- VTAM®
- WebSphere®
- z/Architecture®
- z/OS®
- z/VM®
- z/VSE®
- zPDT®
- zSeries®

The following terms are trademarks of other companies:

- Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

- Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

- Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

- UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.
Preface

This IBM® Redbooks® publication introduces the IBM System z® Personal Development Tool (zPDT®), which runs on an underlying Linux system based on an Intel processor. zPDT provides a System z system on a PC capable of running current System z operating systems, including emulation of selected System z I/O devices and control units. It is intended as a development, demonstration, and learning platform and is not designed as a production system.

This book, providing specific installation instructions, is the second of three volumes. The first volume describes the general concepts of zPDT and a syntax reference for zPDT commands and device managers. The third volume discusses more advanced topics that may not interest all zPDT users. The IBM order numbers for the three volumes are SG24-7721, SG24-7722, and SG24-7723.

The systems discussed in these volumes are complex, with elements of Linux (for the underlying PC machine), IBM z/Architecture® (for the core zPDT elements), System z I/O functions (for emulated I/O devices), and IBM z/OS® (providing the System z application interface), and possibly with other System z operating systems. We assume the reader is familiar with the general concepts and terminology of System z hardware and software elements and with basic PC Linux characteristics.

The author

This series of IBM Redbooks publications was produced by the zPDT development team, with assistance from many other people.

Bill Ogden is a retired Senior Technical Staff Member at the International Technical Support Organization, Poughkeepsie. He enjoys working with new mainframe users and entry-level systems.

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

Keith VanBenschoten, IBM Poughkeepsie, was very helpful in establishing installation and startup processes for the 1090 and in providing test systems.

Theodore Bohizic, IBM Poughkeepsie, helped us understand command, design, and internal details.

Now you can become a published author, too!

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.
Comments welcome

Your comments are important to us!

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

- Use the online **Contact us** review Redbooks form found at:
  ibm.com/redbooks
- Send your comments in an e-mail to:
  redbooks@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:
  http://www.facebook.com/IBMRedbooks
- Follow us on twitter:
  http://twitter.com/ibmredbooks
- Look for us on LinkedIn:
  http://www.linkedin.com/groups?home=&gid=2130806
- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:
- Stay current on recent Redbooks publications with RSS Feeds:
  http://www.redbooks.ibm.com/rss.html
Chapter 1. Linux and zPDT installation

The zPDT system operates as a normal Linux application. The “supported” bases for zPDT, both hardware and software, are described in Volume 1 of this series of documents.

We strongly recommend that the general procedures described here be followed for initial zPDT usage. After you have gained some experience with zPDT, you can explore other installation and usage arrangements. Many of our choices are arbitrary and simply reflect our preference for a simple Linux.

We assume the PC is dedicated to Linux, with the zPDT application used when wanted. If this is not the case, you might consider alternatives:

- For a laptop, you might obtain a separate hard disk for Linux. Changing the hard disk drive in most laptop machines is quick and easy.
- You might consider a dual boot environment, to alternate between Microsoft Windows and Linux.
- If zPDT usage represents a substantial investment for you, consider purchasing a separate machine for it. The cost of a modest PC is often insignificant when measured against the overall investment in System z usage.
- Virtual environments are discussed in Chapter 15 of the third document in this series (SG24-7723-05 or later).

For some of our examples, we elected to install Linux with fixed IP addresses, with firewalls and other security elements disabled. This was to ease communication in a private LAN environment (connected to a small, personal router). Your needs might be different. The zPDT functions are not related to these controls, except that you might need to open firewall access for TCP/IP connections to the zPDT functions. A number of different LAN and TCP/IP configurations are explored in Chapter 3, “LANs and TCP/IP” on page 25.

Ordering requirements

This document does not provide detailed ordering information for zPDT. The ordering process might differ for various categories of users and for different countries. Whatever ordering process is used should result in the following:

- One of the zPDT-supported Linux distributions (specified versions of Red Hat Enterprise Linux (RHEL) or SUSE Linux Enterprise Server (SLES)). These and alternatives are
discussed in Volume 1 of this documentation series. This base Linux is not supplied by IBM. It might need to be ordered, or it might be downloaded from a website.

- A zPDT token (which might need to be activated through an IBM business partner, zPDT supplier, or through IBM ResourceLink). (An alternative to individual tokens exists. This is discussed under “License & serial number servers” in the third volume of this documentation series.)

- The zPDT software, which must be installed before the token can be activated (if it is not supplied already activated). The base zPDT software does not include any System z operating systems.

- Whatever System z software you plan to use, in a format usable with zPDT. This might require a different ordering process than ordering zPDT itself.

Much of the material in this document assumes that you will install the z/OS AD-CD system. If you are installing different System z software, you need to obtain specific instructions for zPDT from the supplier of that software.

**Installation overview**

A summary of the usual installation sequence is this:

1. Read this series of books. You might not remember all the details at this stage, but skim through most of the material before starting. Remember to work as root when indicated and as another userid (ibmsys1 in all our examples) when indicated. If you are new to Linux or zPDT, install a simple system first, before attempting something more complex. Do not use root for all installation and operation actions.

2. Think about your disk partitioning, especially if you plan to install major applications in addition to the zPDT package or if you have especially large PC memory.

3. Gather the required software packages:
   - Linux for your base PC. zPDT checks for Red Hat or SUSE indicators and will not install with other Linux distributions.¹
   - The zPDT software (which might be obtained on a CD or DVD, or by a download).

   Two zPDT packages are available, one for 1090 tokens and one for 1091 tokens. The proper package must be used with the proper token.

   For each package, two prerequisite modules, the zPDT modules, and the license agreement are all in a single (non-rpm) file. The file contains both the Red Hat and SUSE versions of the zPDT code. The proper version is automatically installed on your system.

   - Your System z software (z/OS, IBM z/VM®, or IBM z/VSE®) in a format usable with zPDT.

4. Follow the installation steps described later in this chapter:
   a. Install Linux.
   b. Install x3270 (or another 3270 emulator) if it is not included in your base Linux distribution. Optionally, customize the x3270 keyboard.
   c. Create group zpdt and userid ibmsys1.²
   d. Install the zPDT package.
   e. Customize two Linux files (sysctl.conf and .bashrc).
   f. Copy or create a devmap.

¹ The zPDT installation program checks the base Linux system for Red Hat or SUSE indicators. It is possible that other distributions might have one of these indicators, allowing zPDT installation to proceed.

² This is not required. However, all our examples assume that the zPDT is operated through Linux userid ibmsys1. Whatever userid is used must be no longer than eight characters. Do not attempt to operate zPDT while working as root.
5. Activate your zPDT token if necessary, as described in 1.5, “Token activation and renewal” on page 11. You cannot do this until the zPDT package is installed. (This step might have been done by a business partner or zPDT service provider.)

6. Generally following Chapter 2, “AD CD z/OS installation” on page 15, install z/OS or other System z software:
   a. Select the AD-CD distribution (or another System z operating system)
   b. Unzip the volumes.
   c. Customize or create a devmap.
   d. Start zPDT and IPL your operating system.

7. After you have run a basic system, you might consider LAN configurations.

As is often true with new hardware and software combinations, remember that a given Linux level might not support the newest PC hardware. This is most often seen with new LAN adapter chips and with new graphics chips and/or display panels. Support for these might require additional Linux drivers or upgrades. If you have a new PC model, or an unusual configuration, you must verify that your Linux version is completely operational on your hardware.

1.1 Disk planning

During Linux installations we usually create three partitions on the hard disk:

- A root partition for Linux (including the various zPDT files)
  For a typical laptop, we usually make this about 10 - 20 GB although this is larger than routinely needed. This partition contains all the normal Linux root directories, such as /usr, /lib, /home, /etc, and so forth. If you have additional major applications installed (other than basic Linux functions), this partition might need to be much larger. There should be sufficient free space to accept one or more large core image files.

- A swap partition for Linux. We suggest 4 GB (or larger)
  A common recommendation is (real memory size) + 2 GB, although this might result in some wasted disk space.

- A large partition for emulated System z volumes
  We mount this partition as /z. We normally use all the remaining space on the disk drive for this partition.

If you create a separate /home file system, it should be large enough for several sizeable core image files. At least several gigabytes are suggested.

This suggested disk usage layout is not required. You could make many partitions for the various standard Linux directories. You could place emulated volumes in various directories under /home. You could place emulated volumes in /tmp, and so forth. We recommend our disk layout as a starting point solely because it is simple and it isolates emulated System z volumes from the normal Linux files. This isolation is useful if you reinstall Linux (without disturbing your emulated volumes) and it might have minor performance benefits because it tends to reduce fragmentation in the disk space used for large emulated volumes.

3 A machine with larger memory will typically have a larger Linux file system. Among other things, it should be able to hold one or more core image files that might be created in unusual situations. A core image file created under zPDT might be somewhat larger than the System z size defined in the devmap.

4 Core images (which are a debugging tool) should not be encountered often, but they might be needed for zPDT service in the rare event of serious problems with zPDT.
If you plan a dual boot system, then you will have at least one more partition (probably for Windows). You might need to shrink the existing Windows partition to make room for the new partitions. The various ways to do this are not covered in this document.

1.2 Linux installation

Install your Linux distribution. You might select only those packages that are needed for basic Linux operation, or you might install everything in your distribution. Consider the following requirements and suggestions:

- You must include Linux 32-bit support, which is an option during Linux installation. It is required for the zPDT token drivers. (On RHEL 6.x systems you need the libstdc++-xx.xx.i686.rpm, not the compat-libstdc rpm. On RHEL 6.0 the specific name is libstdc++-4.4.4-13.el6.i686, but this might be updated for later releases. In the Add/Remove Software function (during RHEL installation or later updates) there is a small box that shows up after clicking Filters on the menu bar, with the label Only Native Packages. This was checked by default. You should uncheck it to see the i686 rpms.)

- Select Universal Time (UTC) for your base PC, if this is possible. (This might not be possible if you also run Windows on the same PC. Do not consider changing the PC time-of-day when switching between Linux and Windows, because this can disrupt your zPDT usage.)

- Although not a Linux option, machine hiperthreading (if available) must be disabled at the BIOS level. z/OS slowdowns might occur at random times if you have hiperthreading active when running zPDT with multiple CPUs.

- For Red Hat releases we recommend including the dmidecode rpm. For SUSE releases we recommend including pmtools. Both packages include the upddecode tool. These packages are optional, but they might be used to provide additional diagnostic information if there is a problem with zPDT. If they are not in your Linux distribution, do not worry about it.

- We sometimes use gnome as our desktop manager, and this is reflected in our examples. With the advent of gnome 3, we sometimes use the Xfce desktop (which may be selected among the other desktop choices during Linux installation). The Xfce interface is more like gnome 2. You could also select KDE (we have no indication that it would not work with zPDT). zPDT is not sensitive to the desktop manager.

- We usually install telnet server if it is present in the Linux distribution, although this is an insecure service. It is not necessary to use it, but it is convenient to have it installed if a simple telnet session to the zPDT machine is needed. (All zPDT functions can be run remotely, through telnet or ssh or other packages.)

- Install x3270 if it is included in your Linux distribution. It is not installed by default. You might need to search diligently to find it, or to determine that it definitely is not present in your Linux distribution. Generally, we find that the Linux installer provides a way to specify installation packages at the rpm level, but some effort might be needed to find this path.

- Install vsftpd or some other Linux ftp package. It is not needed for zPDT usage, although you might want to transfer files between Linux and z/OS using ftp.

- We have sometimes found it advisable to perform an online update for the Linux distribution. If this is available for your Linux, we suggest you perform the update.

- Determine if additional drivers or driver updates are needed for new hardware on your PC.

- You must manage whatever firewall and other security functions that you install with your Linux. We suggest initially disabling any firewall when first working with zPDT. After you are familiar with zPDT operation, you can reestablish the firewall functions. If you have
external TCP/IP connections (for local 3270 connections, for OSA connections, or for CTC connections) you must provide appropriate port holes in any firewall you use.

### TN3270e clients

IBM has used these TN3270e clients with the recent zPDT offerings:

- x3270 (recent versions)
- Recent PCOMM releases (running on Windows systems)

We most commonly use x3270. Our last step during many Linux installations has been to install x3270, because it was not included with most current Linux distributions. An x3270 package is usually a single rpm, such as:

```
x3270-3.2.20-467.1.x86_64.rpm
```

Other x3270 levels may be used, or another 3270 emulator may be used.

#### 1.2.1 x3270 keyboard maps

The default x3270 keyboard assignments are not in the traditional 3270 style. In particular, the large Enter key on the PC keyboard functions as the 3270 Enter key. With traditional 3270 keyboards this same key provides a new line function and the 3270 Enter key is located where the right-side Ctrl key is located on most PC keyboards.

There is no requirement to change the default x3270 keyboard mapping. If you prefer the more traditional mapping, use the following steps (working as root):

```
# cd /usr/share/X11/app-defaults
```

This directory might contain file `X3270` (with an upper case letter X). Use the appropriate path to app-defaults for your Linux and verify that file `X3270` is present. If it is not present, you can consider using a local x3270 profile, which is described later.

```
# gedit X3270
```

(use your favorite editor)

(scroll to the stanza named `X3270.keymap.base.3270: #override`)

(scroll to second line in this stanza):

```bash
...
Shift<Key>Return:   Newline()\n
<Key>Return:       Enter()\n
<Key>Linefeed:     Newline()\n
<Key>Backspace:    Erase()\n
<Key>Control_R:    Enter()\n
<Key>Control_L:    Reset()\n
<Key>Return:       Newline()\n
<Key>Pause:         Clear()\n
<Key>BackSpace:     BackSpace() Delete()\n
<Key>KP_Enter:      Enter()\n
<Key>End:          EraseEOF()\n
<Key>Prior:         PF(7)\n
<Key>Next:         PF(8)\n
Shift<Key>Tab:     BackTab()\n
...
```

---

5 An exception is SLES 11, which includes x3270.

6 Do not place any blank lines, tab characters, or extra blanks at the end of the lines within these definitions! The `X3270` file appears to be sensitive to unexpected characters within the definitions. Some Linux distributions do not contain this file and customizing x3270 for these distributions can be more challenging.
Notice that the 3270 screen defaults to model 4 (with 43 lines). Our ThinkPad keyboard contained extra keys associated with Microsoft Windows usage, making the left and right Ctrl keys smaller than they are on some keyboards. We did not attempt to map these Windows keys to any 3270 function.

As a result of these changes to the keymap, common 3270 functions are as follows:

<table>
<thead>
<tr>
<th>Function</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>3270 Enter</td>
<td>Right-hand Ctrl key or the numeric keypad Enter key</td>
</tr>
<tr>
<td>3270 Reset</td>
<td>Left-hand Ctrl key or alt-r</td>
</tr>
<tr>
<td>3270 Clear</td>
<td>Pause or alt-c</td>
</tr>
<tr>
<td>Next line</td>
<td>Large Enter key on keyboard</td>
</tr>
<tr>
<td>PA1</td>
<td>alt-1</td>
</tr>
<tr>
<td>PA2</td>
<td>alt-2</td>
</tr>
<tr>
<td>F13</td>
<td>shift-1</td>
</tr>
</tbody>
</table>

You can adapt these instructions to other x3270 versions or simply use the default keymap distributed with x3270.

**Local x3270 profile**

An alternative to changing the app-defaults/X3270 keyboard map (or, if this file does not exist) is to override the x3270 default keyboard using a profile in your home directory. To do this, create a file (in your home directory) named `.x3270pro` (note the period as the first character of the file name):

```bash
x3270.keymap: bill
! Use Bill's overrides
! Define the overrides
x3270.keymap.bill: #override
  <Key>Control_R:  Enter()
  <Key>Control_L:  Reset()
  <Key>Return:     Newline()
  <Key>Pause:      Clear()
  <Key>BackSpace:  Delete()
```

It seems that x3270 keyboard files are sensitive to extra spaces and tab characters. Do not have anything after the `\n\` in the text lines. In this file, we spaced the `<Key>` field starting in column 4, although this was arbitrary.

**x3270 fonts**

If x3270 is installed from a separate source, it might not have its normal fonts. In the x3270 fonts menu there might be an option for iso fonts. We selected the following one:

```
-eti-fixed-bold-r-normal--18-180-72-72-c-90-iso8859-1
```

The 18 that is embedded in the name is the point size. A similar choice, with 24 in this position, selected a larger font.

### 1.2.2 Other Linux notes

Always use the same Linux userid for zPDT operation. This Linux userid must be no longer than eight characters. Multiple zPDT instances, as described in the third book in this series, require a different Linux userid for each instance.

---

7 The Linux userid is used as the LPAR name under zPDT, and LPAR names are limited to eight characters or less. (Only a subset of LPAR-like functions are provided by zPDT.)
We found the following Linux commands to be useful for setup verification:

$ ulimit -a  
(display various limits for this user session)

$ ipcs -l  
(lower-case l; shared memory limits)

$ /sbin/sysctl -a  
(many kernel and other system parameters)

$ /sbin/ifconfig  
(LAN status)

Other Linux distributions
The Linux distributions supported by zPDT are discussed in the first book in this series (IBM publication SG24-7721-05). zPDT is not designed to run on all Linux distributions. “Force fitting” zPDT to other distributions might create obscure errors.

CD/DVD drive access
For the Linux distributions we used, the path used to access the CD/DVD drive is /media/xxxxx or /run/media/ibmsys1/xxxxx where xxxxx is the volume name or title of the CD/DVD. Whoever creates a CD/DVD can assign a title. You might be able to determine a CD/DVD title by observing the window title that is presented when the Linux automount function detects the new CD/DVD.

Linux PATH
We suggest that you do not add other directories before /usr/z1090/bin in the PATH and LD_LIBRARY_PATH variables. There are many commands and executed modules provided with zPDT, and these correspond to Linux file names that are accessed through the PATH variables. For example, the command d is used to display System z memory. If you place another directory containing a file named d earlier in the PATH, then the zPDT d function will not be available in the normal manner. Various internal zPDT functions assume they can access zPDT modules through PATH and LD_LIBRARY_PATH, so you must ensure that this is possible.

System time
We suggest setting your hardware clock to UTC time. This avoids problems when daylight saving time starts and stops. The zPDT token is sensitive to the hardware clock time and will not operate if the time appears to move backward. If the machine is shared with another operating system that expects local time (instead of UTC time), you might experience a one-hour non-operational time when shifting from daylight saving time to standard time.

1.3 Install zPDT software

Decide on the Linux userid you want to use for zPDT. We use ibmsys1 in all our examples, but you can select any userid that is eight characters or less. We also create a Linux group named zpdt and add userid ibmsys1 to it. A special group for zPDT is not required, but it can be used for additional security controls.

Note that $ prompts (throughout the examples in this document) indicate a non-root userid, and # prompts indicate we are working as root. We recommend that you always log in as ibmsys1 and then use an su command to switch to root when needed. The following

---

8 This and the associated “time cheat” messages are discussed in the third volume of this series.
9 The third volume contains a note about limiting access to the only zPDT module that involves running with root authority.
10 There is nothing special about userid ibmsys1. We consistently use it to illustrate zPDT operation. Any userid with 8 or fewer characters can be substituted for ibmsys1, but that userid should be consistently used for all zPDT installation and operation actions.
directions assume that a single zPDT instance will be used. (Multiple zPDT instances require multiple userids, such as ibmsys2 and ibmsys3.)

If you have not already done so, create group zpdt and user ibmsys1, which should be a member of group zpdt. By default, userid ibmsys1 has /home/ibmsys1 as its home directory and most zPDT control files appear in subdirectories here. We created file system /z as a separate partition during our Linux installation.11 We want userid ibmsys1 to own this file system:

(log on as ibmuser1)
# su
# chown ibmsys1:zpdt /z

A single executable file is used to install the zPDT software.12 The file name changes with maintenance releases, but has the following general format: 13

z1090-1-4.45.23.x86_64
z1091_xxx-1-4.45.23.x86_64

The single file contains the following:

- An sntl-sud rpm at the correct level (A driver for the zPDT token)
- A zpdt-shk-server rpm at the correct level (Another token program)
- The primary z1090 or z1091 rpm for SUSE
- The primary z1090 or z1091 rpm for Red Hat
- An installer program that displays a license and then installs the rpms. The correct rpm (Red Hat or SUSE) is automatically selected for your base Linux system.

Proceed with zPDT installation as follows. The first goal is to move the installation file to a convenient directory, such as /tmp. If you obtained the zPDT installation file through ftp or another download method, you might have already placed it in /tmp.

(log on as ibmsys1)
$ su
# cd /media/ROM
# cp /media/ROM/z1090-1-4-45.23.x86_64 /tmp
# cd /tmp

Execute the installer program, for example: 15

# chmod u+x z1090-1-4.45.23.x86_64
# ./z1090-1-4.45.23.x86_64

Scroll through the license that is displayed and reply to the question at the end. The various rpms are then installed automatically. The zPDT installer program performs the following tasks, removing older versions of these programs as needed:

- Two prerequisite rpms are installed.
- The z1090 or z1091 rpm is installed, mostly in /usr/z1090/bin. (/usr/z1091/ is not used)
- A set of man files is loaded into /usr/z1090/man.
- A /usr/z1090/uim directory may be created.

11 This might not be the case for the IBM Open Client, but we ignore this exception here.
12 This file differs for 1090 and 1091 systems.
13 RDzUT users might have file names beginning with something like “z1091_801” instead of “z1090”. Use whatever file name is provided with your package.
14 This example assumes that ROM is the title of the CD. You must determine the title of your CD.
15 If you are coming from an older release of zPDT you might have multiple copies of sntl-sud or zpdt-shk-server installed and the installation step will fail. In this case, do the following:
   # rpm -e z1090
   # rpm -e --allmatch sntl-sud
16 The “./” characters before the file name tell Linux to execute this file from the current directory.
**Installer options**

The installer program has three optional functions. Using the file name in the preceding example, the functions are listed here:

```
# ./z1090-1-4.45.23.x86_64 --refresh   (reinstall current zPDT level)
# ./z1090-1-4.45.23.x86_64 --refreshall (reinstall zPDT and prerequisites)
# ./z1090-1-4.45.23.x86_64 --removeall (remove zPDT and prerequisites)
```

The prerequisites mentioned here are two rpms that are needed to access the USB token.

1.3.1 Alter Linux files

You must alter two Linux files before you can use zPDT. The first alteration is to `/etc/sysctl.conf` and involves changing a number of Linux kernel parameters. The second alteration is to the `.bashrc` file in your home directory; this adds the zPDT directories to your userid's PATH variable.

You can manually edit the relevant Linux files or use zPDT commands to make the changes. The zPDT commands are listed here:

```
# /usr/z1090/bin/aws_config   (You must be root to use this command)
$ /usr/z1090/bin/aws_bashrc   (You must not be root to use this command)
```

The complete path name might be used for these commands (as shown here) because the Linux PATH might not yet include the zPDT files.

If you want to manually edit and add lines to `/etc/sysctl.conf`, we indicate the use of `gedit` but you may use any suitable editor (such as `vi`) to add the indicated lines. Unfortunately, some newer Linux versions restrict the `gedit` editor such that it is not usable when running as `root`. You might need to overcome this; one way is to change the permissions or ownership of the target files, using the `chmod` or `chown` commands. We ignore this detail in the following examples.

Some Linux distributions already have acceptable values for `shmmax`, `msgmnb`, `msgmax`, and `core_uses_pid`, but other distributions may need to have all these values set.

```
# gedit /etc/sysctl.conf   (the following lines should begin in column 1)
kernel.shmmax=1800000000000000
kernel.shmall=12000000
kernel.core_pattern=core-%e-%p-%t
kernel.core_uses_pid=1
kernel.msgmni=512
kernel.msgmax=65536
kernel.msgmnb=65536
net.core.rmem_max=1048576
net.core.rmem_default=1048576
# /sbin/sysctl -p /etc/sysctl.conf
```

Notes for sysctl values

The `shmmax` value shown establishes the maximum shared memory segment size that a user can request. All system z memory, plus several other zPDT work areas, are in Linux shared memory segments. The `shmmax` value should be 10-20% larger than the System z memory defined for the largest zPDT instance you use. The example shown (18000000000000 bytes)

17 Some versions of the IBM Open Client reset these values when maintenance is applied. If this happens, you should again enter the values shown here and run `/sbin/sysctl`.

18 We suggest that you do not attempt to use `vi` unless you have a basic familiarity with it.
would be suitable for a zPDT instance with up to 14-15 GB memory defined. There is no need to attempt an exact fit for the shmmax number; the example here may be used unless your define System z memory is larger than, for example, 15 GB.

Another parameter, kernel.shmall, sets the total shared memory size of all users. The value of shmall is specified in units of page size, which is usually 4096. The default value of shmall is usually quite large and acceptable. However, if you have multiple zPDT instances, all with large System z memory, you might exceed the default shmall value. If this happens you need to include a parameter such as:

```
kernel.shmall=12000000  # or larger, if appropriate
```

This would result in the total amount of shared memory, for all users, to be 12,000,000*4096 or about 48 GB. This value should be greater than the number of zPDT instances times the System z memory size for each instance plus about 10-20%. A number much larger than needed appears to do no harm.

Remember: the shmmax number is for bytes, and the shmall number is for pages.

The kernel.msgmni number, specified as 512 in this example, might need to be larger if you have many emulated I/O devices. The msgmax and msgmb changes are not needed for some Linux releases because these are the default settings. However, including these parameters in sysctl.conf does no harm; the settings are needed for proper OSA operation.

The net.core parameters might be needed if Ethernet large frames are used. These seem to do no harm, so you may always include them. In this context, any frame with more than 1500 bytes is considered large.

**Notes for .bashrc**

The .bashrc file is changed, as follows:

```
# exit  # (leave root)
$ cd /home/ibmsys1  # (my login directory)
$ gedit .bashrc  # (use your favorite editor)

Add the following lines beginning in column 1):

export PATH=/usr/z1090/bin:$PATH
export LD_LIBRARY_PATH=/usr/z1090/bin:$LD_LIBRARY_PATH
export MANPATH=/usr/z1090/man:$MANPATH
ulimit -c unlimited
ulimit -d unlimited
ulimit -m unlimited  # (if more than 128 emulated I/O devices)
ulimit -v unlimited  # (if more than 128 emulated I/O devices)
```

Double-check the entries in these two Linux files. Errors here might be difficult to detect later. The ulimit -m and -v statements are not required for most users and should probably be excluded unless you have more than 128 emulated I/O devices.)

**Other files**

Check your zPDT distribution materials to see if there are sample devmap files that might be helpful. Copy these to /home/ibmsys1, for example:

```
$ cd /tmp/ROM
$ cp aprof13 /home/ibmsys1/aprof13  # (sample devmap)
$ chmod 664 /home/ibmsys1/aprof13
```

---

19 The default value on several distributions is 1,152,921,504,606,846,720, which is huge. However, on at least one IBM internally-used Linux the default is much lower and may need to be adjusted.

20 These might be provided by your zPDT service provider.
Note that any sample devmaps probably must be edited to match your configuration and file names. If you do not find a sample devmap, you need to create one before you can start zPDT.

**Reboot Linux**

Reboot Linux to pick up all the changes you have made. Then use the `z1090instcheck` command to partly verify your environment for running zPDT. Your new PATH is needed to find the command:

```
(log in as ibmsys1)
$ z1090instcheck (the same command is also used for 1091 systems)
```

If this command is not found, you do not have the PATH variables set or you did not install the zPDT code correctly. Note that this command does not check any devmaps that you may have defined or copied.

### 1.4 zPDT serial numbers

In the simple case, in which you have a single zPDT token that is used only on your PC, the System z serial number (when you start zPDT operation) is derived from the token serial number of the token.

In more complex environments, the System z serial number used may be set in other ways. This is described in detail in the chapter titled “License & serial number servers” in the third volume of this documentation series.

### 1.5 Token activation and renewal

**Note:** At the time of writing, the material in this section (about token license renewals) applies primarily to IBM internal users and a few special situations. It does not apply to tokens obtained through normal customer channels. It does not apply to 1091 tokens.

A USB hardware key (token) is normally valid for a year from the time it was last activated. Activation (and lease renewal) might be handled by your zPDT service provider (such as an IBM Business Partner) or by using IBM Resource Link® in some cases. IBM Resource Link usage for token activation requires an IBM employee userid or a userid that has been added to a list of certain zPDT token users.

Copy the information that is printed on the token tag (illustrated in Figure 1-1) attached to the USB hardware key. Save this information in a safe place.

---

21 This was true at the time of writing. Future availability plans may manage this in a different way.
The data in the 11S and MTSN fields on the tag is required to:

- Activate the USB hardware key for the first time.
- Extend the lease for one year.
- Activate and use a replacement USB hardware key.

A Resource Link profile (userid) is needed. This may be an IBM employee profile or a PWD-approved profile for other users. If you do not have one, go to the following link and follow the Register for an IBM ID link (in the upper right part of the panel):

http://www.ibm.com/servers/resourcelink

After establishing a userid and password, follow the Sign in link to create your Resource Link profile. Wait two hours after creating the profile before signing back in. The profile information must be replicated among several servers, which can take some time.

**Activation (or renewal, date extension, or lease extension)**

After this preliminary work, the key can be activated, renewed, or have the lease date extended (these are all provided in the same way):

1. Connect the USB hardware key to your zPDT system, using any USB port. (You must have already installed the zPDT software.)

2. Working as `root`, create a request file using the Secure Update Utility:

   (log in with a normal zPDT userid, such as `ibmsys1`)

   ```
   $ su  # change to root
   # cd /usr/z1090/bin  # must be in this directory
   # SecureUpdateUtility -r myrequest  # myrequest is an arbitrary file name
   # exit  # leave root
   ```

3. If necessary, move this request file to the computer used to access Resource Link and log on to Resource Link. (The file name will have `.req` added as the name extension.)

4. On Resource Link, navigate to **Tools → 1090 Support → Date Extension** and enter the data from your hardware key tag. Use the last six digits of the 11S field. The serial number (the MTSN field) can be entered with or without the dash; it is not case sensitive. Enter the file name of your request file. Finally, click Submit.

5. Resource Link will create an update file and send it to you by email. (This typically takes about 10 minutes.) Receive this file and move it to your zPDT machine, if necessary. The file name will be the same name that you sent, but with `.upw` as the name extension.

6. Apply the file to the USB hardware key:

   ```
   $ SecureUpdateUtility -u myrequest.upw
   ```
7. After the Secure Update is successfully applied, unplug the USB hardware key. Wait at least 10 seconds and then reconnect the hardware key. It is now ready for routine zPDT operation.

1.6 Installing a new zPDT release

New zPDT releases are typically available through your Business Partner or (for IBM employees) through Resource Link. The installation procedure is the same regardless of the source. Installation is exactly the same as described earlier.

A summary of the steps is as follows:

1. Obtain the new distribution file.

2. Working as *root*, execute the distributed file. It will delete the previous release and install the new release. The process takes only a few seconds and does not disturb any of your customization.
AD CD z/OS installation

The System z Personal Development Tool provides System z functionality and associated utility programs. It does not include any System z software. System z software, including operating systems, utilities, middleware, applications, and so forth, must be obtained separately.

For software licensing purposes, a zPDT system is a System z and all software licensing requirements that apply to a larger System z installation also apply to a zPDT installation. This statement applies to all System z software from IBM and, we assume, applies to all System z software available from other vendors.¹

The discussions in this chapter (and elsewhere in this series) assume that proper licenses have been obtained for the System z software. Licensing arrangements (and associated costs) can be complex topics and are not further addressed in this document.

**Important:** The discussions in the remainder of this document and in the third volume in this series assume the reader has a general familiarity with z/OS systems programming and understands how to access various control data sets. We highlight specific details that might be relevant to zPDT usage and the current AD-CD releases. This is not intended as an introduction to z/OS administration.

Furthermore, we assume basic familiarity with the AD-CD z/OS package. Find update information about the AD-CD packages at the following URL:

http://dtsc.dfw.ibm.com

¹ In this discussion we include software for IBM zSeries®, S/390s, and so forth, in the general category of System z software.
2.1 General principles

All current IBM System z operating systems (assuming proper licenses exist) are supported for zPDT usage. This includes current versions of z/OS, z/VM, and z/VSE. Linux distributions intended for System z usage may be used, but all functions and configurations have not been extensively tested. Older versions of operating systems and other software might work correctly (if they are at least at the XA level), but there is no formal testing or support for older software.

Software installation methods might be different for zPDT systems than for traditional System z installations. This difference is due to the differences in I/O devices available on zPDT systems and on larger System z machines.

2.2 System z operating systems

There are specific limitations for installing IBM operating systems. These limitations are related to the use of the software media and packaging techniques involved and are not limitations on the use of the operating systems, after they are installed.

The most common limitation is for software that is distributed on tape. To install this software, your zPDT system must have a tape drive and these are not commonly available for PC machines. Another limitation is related to any System z software that is packaged in such a way that installation requires specific System z HMC functions.

2.2.1 Media

In most cases (when a tape drive is not available) installation media is limited to CD, DVD, and LAN connections. (We can consider FTP as “media” in this context.) CD or DVD files must be in formats that can be processed for zPDT. There are two meaningful formats:

- A Linux image of an emulated 3390 drive\(^2\) that can be restored in the 3390 format used by zPDT; the image might be compressed (using gzip, for example) and would need to be uncompressed before use with zPDT. Likewise, the image might be in tar files and would need to be untarred (and possibly uncompressed) before being used with zPDT.

  The 3390 drive image format must be produced by another zPDT system, because no other product uses the same 3390 image format that is used by zPDT. Whatever preliminary unpacking or uncompression is needed must be done by Linux utilities before the 3390 image can be used by a System z operating system.

- A tape image in awstape format; such images appear as “real” tape volumes to System z operating systems operating on zPDT, and can be processed as such by using emulated tape drives. The tape might contain product installation material (in SMP/E format, for example), an ADRDSSU dump of a disk volume, or any other tape data usable by System z programs.

The same files, in the formats described, could be exchanged by FTP (to the base Linux of the zPDT system) instead of using CD or DVDs. This would not affect the processing requirements.

Another media option is to FTP a product (or other data) directly to z/OS. Some System z software is distributed in this format. zPDT must have a running z/OS and LAN connection to use this method. (Most of our discussion is for z/OS, but z/VM, z/VSE, or Linux for System z

\(^2\) 3380 images could also be used, but we ignore these here.
might be used in the same way. The point is that a working System z operating system must be installed before additional software can be sent directly to it through FTP.)

It is important to differentiate the handling of these methods:
- CDs and DVDs must be processed by Linux programs (unless they contain awstape files).
- awstape files must be processed by z/OS (or another System z operating system), although the transport of awstape files can be managed by Linux through CD/DVDs or FTP.
- Direct FTPs to z/OS might be in other formats, for example in formats suitable for processing by SMP/E or the TSO XMIT command. In any event, these are System z formats and not Linux formats.
- An emulated 3390 volume (after decompression, if necessary) is a large Linux file that is meaningful only to System z software.

2.3 Packages for zPDT

At the time of writing, organizations within IBM produce one z/OS package that can be directly installed on zPDT systems. This is the Application Development System.³ It is distributed on DVDs or by downloads. These packages are not available to the general public; you must be properly licensed to obtain or use these packages.

This package is not part of the IBM System z Personal Development Tool. It is a convenience package for users of this tool. It is a z/OS system, with additional System z software products, and requires appropriate System z software licenses for use. The AD-CD system is intended for use by members of the IBM PartnerWorld® for Developers, among other users.

Other packages, generally grouped under the AD-CD name contain z/VM, z/VSE, and a z/OS IBM Parallel Sysplex® starter system. Separate zPDT license features or other agreements might be needed to obtain these packages.

2.4 Installing an AD-CD system

The following examples use volsers corresponding to the z/OS 1.13 (January 2012 and January 2013) release of the AD-CD system. These volsers tend to change in a standard pattern for new releases.

Attention: There is a known problem with the z/OS IEBCOPY utility shipped with z/OS 1.13. The problem is resolved with PTF UA67459. An alternative to applying this PTF is discussed in Chapter 3 of the third book in this series (SG24-7723). Without remedial action (PTF or alternative action), a partitioned dataset might become unusable after being compressed with IEBCOPY.

At the time of writing, we do not know whether the PTF will be integrated in z/OS 2.1.

2.4.1 Specific installation instructions

There are multiple DVDs for a z/OS AD-CD release. The first DVD contains the 3390 volumes needed to IPL and to use common subsystems. It might also contain a README file, sample

³ This is also known as the AD-CD system, where CD means Controlled Distribution.
The distributed emulated volumes are all in gzipped format. The z/OS 1.13 AD-CD system uses DVD titles such as DVD1, DVD2, and so forth; again, we stress that you must determine the volume title on each DVD. (Some CD titles containing multiple blanks are difficult to use with a cd command. We used the Linux automatic command completion function with these.) AD-CD system installation might be as follows, assuming our target directory for emulated 3390 volumes is /z:

```
(work as userid ibmsys1)
$ cd /run/media/_userid/DVD1
$ gunzip -c zdres1.gz > /z/ZDRES1
$ gunzip -c zdres2.gz > /z/ZDRES2
$ gunzip -c zdsys1.gz > /z/ZDSYS1

And so forth for all the volumes to be installed.
```

We elected to use the volser of the 3390 volume as the Linux file name that holds the volume. We use upper case letters simply to make these emulated volume file names more distinctive. There is no requirement to use the volser as the Linux file name, and there is no requirement to use upper case names.

The files containing emulated volumes (and the directory containing these files) must have read and write permissions for the userid running zPDT. Assuming use of the ibmsys1 userid, we suggest that all such files and their directories (/z, in our examples) be owned by ibmsys1.

**File name considerations**

A 3390 volume exists in a single Linux file. For example, a 3390-3 volume exists as a 2.8 GB (approximately) Linux file. A 3390 volume has a volume serial (volser) that is written in the first track of the 3390 volume. The Linux file holding the (emulated) 3390 volume has a Linux file name, which is specified in a devmap. There is no required relationship between the volser and the Linux file name. For example, volser WORK02 might be in /tmp/mysys/ckd001.

We strongly suggest that you make the Linux file name reflect the volser, if at all possible. For example, volser WORK02 might be in /z/mysys/WORK02 or /z/mysys/WORK02.ckd. When assisting zPDT users with various problems, we (the zPDT team) have wasted much time (both our time and user time) working with the wrong volumes or determining which Linux file contains which 3390 volume. A Linux naming convention that reflects the volser can avoid wasted debugging time.

### 2.4.2 IODF device numbers

We must know the device numbers (“addresses”) used by the z/OS system. (These may be changed after the z/OS system is running. Changing involves creating a new IODF data set, new IPLPARM member or members, and reIPLing z/OS.) Most users of the AD-CD system accept the device numbers generated in the IODF supplied with the AD-CD system. These are listed here:

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>DEVICE</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>00C</td>
<td>2540R</td>
<td>Card reader. Useful as an emulated device.</td>
</tr>
</tbody>
</table>

---

4 The volser is written by the ICKDSF utility program (for a new volume), or is already present in a volume that was downloaded or unzipped from a DVD.
00E-00F  1403-N1  Line printers. Useful as an emulated device.
120-15F  3380  Disks. (Control units defined for 120-127)
300-318  3390  3390 disks
400-40F  0SA  OSA
550-55F  3400  Round tape drives
560-56F  3480  Without COMPACT feature
580-58F  3490  Tape drives
590-59F  3590  Tape drives
700  3270  Terminal. AD-CD systems use as NIP & z/OS master console
701-73F  3277  Terminal. Normally for VTAM (TSO, CICS, etc)
900-907  3277  Terminal. Normally for VTAM (TSO, CICS, etc)
908  3270  Could be used as a z/OS console
909-91F  3277  Terminal. Normally for VTAM (TSO, CICS, etc)
A80-AEF  3390  Disks
E20-E23  CTC  Typically for 3172s that appear as CTC devices
E40-E43  CTC  Typically for 3172s that appear as CTC devices

These addresses are all three hex digits. This is due to historical reasons. Both the AD-CD system and zPDT system can work with four-digit addresses. These addresses have been stable for many releases of the z/OS AD-CD system; however, it is possible they might change in future releases. Also, note that the slightly different AD-CD version prepared for 1091 systems may have more 3390s defined at addresses such as 1Axx and so forth.

In principle the 3390 IPL volume, for example, could be mounted at any address in the 300-318 or A80-AEF range. In practice, the IPL volume and the volume containing the IODF and other key data sets are usually mounted at addresses A80 and A82:

<table>
<thead>
<tr>
<th>VOLSER</th>
<th>ADDRESS</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZDRES1</td>
<td>A80</td>
<td>IPL volume and key z/OS libraries</td>
</tr>
<tr>
<td>ZDSYS1</td>
<td>A82</td>
<td>Paging, spool space, LOGGER data sets, VSAM, etc</td>
</tr>
</tbody>
</table>

The other volumes can be mounted at any convenient address that is defined in the IODF as a 3390. In practice, many users simply start at address A80 and increment it sequentially for each additional volume.

### 2.4.3 zPDT control files

Before the AD-CD system can be used, an appropriate devmap must be created. A basic example would be as shown here:

```
$ cd /home/ibmsys1
$ gedit aprofl3

[system]
memory 4000m
processors 1
3270port 3270

[manager]
name aws3274 0002
device 0700 3279 3274 mstcon
device 0701 3279 3274
device 0702 3279 3274
device 0703 3279 3274
device 0704 3279 3274
```
name awsckd 0001
device 0a80 3390 3990 /z/ZDRES1
device 0a81 3390 3990 /z/ZDRES2
device 0a82 3390 3990 /z/ZDSYS1
device 0a83 3390 3990 /z/ZDUSS1
device 0a84 3390 3990 /z/ZDUSS2
device 0a85 3390 3990 /z/ZDPRD1
device 0a86 3390 3990 /z/ZDPRD2
device 0a87 3390 3990 /z/ZDPRD3
device 0a8a 3390 3990 /z/ZDPAGA
device 0a8b 3390 3990 /z/ZDPAGB

#(continue with whatever additional volumes you installed.)

Gaps in the assigned address numbers do not create a problem. The devmap can have any name and be placed in any directory. It is best if it is in the directory you will use when starting zPDT so that you do not need to enter a full path name when using it.

The processors statement determines how many System z CPs are started for this zPDT instance.

We suggest you do not define OSA devices for your initial z/OS startup. The OSA definitions can be a little more complex and we suggest you verify that your basic z/OS system is operational first.

### 2.4.4 IPL and operation

Start zPDT with an `awsstart` command. Among other functions this starts the zPDT device manager that emulates local, channel-attached 3270 terminals. Using the `awsstart` command creates a z1090 subdirectory in the current home directory (if it does not already exist) and a number of zPDT-related directories below it.

```
$ cd /home/ibmsys1
$ awsstart aprof13

(wait for messages. Press Enter to regain the $ prompt.)

AWSSTA014I Map file name specified: aprof13
0 Snapdump incident(s), RAS trace and RAS log files occupy 657046 bytes
in /home/ibmsys1/z1090/logs.
Associated files, logs, and core files occupy 10364 bytes in
/home/ibmsys1/z1090/logs
```

Using the same Linux window (or a different window, if you prefer) start at least two local 3270 sessions:

```
$ x3270 -port 3270 localhost &
$ x3270 -port 3270 localhost &
$ x3270 localhost:3270 &
```

- `x3270` is the name of the program.
- We assigned Linux TCP/IP port 3270 for this function. This is specified in the devmap. The port number is arbitrary, but should not be used for any other purpose in your system. Port 3270 is usually a good choice and is easy to remember.
- We want to connect to our own Linux system; this is indicated by the `localhost` operand.
- The ampersand (`&`) causes the `x3270` program to execute in the background, leaving the Linux window free for additional commands. We can recall and execute the `x3270` command repeatedly to create multiple 3270 sessions.
The 3270 window will display identification lines if there has been no data sent to it by the System z software. These lines indicate the terminal identity by address and LUname or IP address. A number of options are available for working with these LU names and these are discussed in Volume 1 of this series. The File and Options menus at the top of the x3270 window can be used for a variety of functions. Changing the font size (using the Options menu) has the effect of changing the 3270 window size.

The 3270 session for the z/OS console (address 700 for the AD-CD system) should be ready before IPLing z/OS. Next, issue the appropriate IPL command in the Linux window:

```bash
$ ip1 a80 parm 0a82cs
```

After a few seconds, the initial z/OS messages should appear on the 3270 session at address 700. During the first IPL of the AD-CD system (or an IPL after a long period of non-usage, or a changed System z serial number) you might see messages similar to the following:

```
IXC420D  REPLY I TO INITIALIZE SYSPLEX ADCDPL, OR R TO REINITIALIZE XCF
```

If this message is issued, go to the 3270 session displaying the message and enter the following command:

```
r 00,i
```

After IBM VTAM® is started, the VTAM logo should display on the other 3270 sessions.5

There is usually a writeup for each AD-CD release that provides details about different IPL parameters and TSO logon procedures. A brief summary for the z/OS 1.11 AD-CD system is shown here:

<table>
<thead>
<tr>
<th>IPLparm</th>
<th>LogonProcedure</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oa82cs</td>
<td>ISPPPROC</td>
<td>Basic IPL without DB2, etc. Cold start JES, CLPA</td>
</tr>
<tr>
<td>Oa8200</td>
<td>ISPPPROC</td>
<td>Subsequent basic IPLs. Warm start.</td>
</tr>
<tr>
<td>Oa829c</td>
<td>DBSPROC9</td>
<td>Initial IPL for DB2 V9, etc</td>
</tr>
<tr>
<td>Oa829w</td>
<td>DBSPROC9</td>
<td>Subsequent IPLs for DB2 V9, etc</td>
</tr>
</tbody>
</table>

Userid IBMUSER is always present on z/OS and is typically used for initial TSO logons. The password for IBMUSER should be published with any AD-CD documentation. It is typically SYS1 or IBMUSER. If there are security concerns about your system, change this initial password as soon as possible.

The Linux command window that was used for the `awsstart` command should be kept open, if possible. Asynchronous messages from zPDT are sent to this window. You can enter zPDT commands from other windows, but it is possible that you might miss significant messages that are sent to the original window.

### 2.4.5 Shutting down

z/OS should be shut down cleanly, if possible. Enter the command `s shutdown` at the z/OS console and reply to any messages produced. The message `ALL FUNCTIONS COMPLETE` indicates that JES2 can now be stopped with the command `$PJES2`. After JES2 ends, System z operation can be stopped. The zPDT system is stopped with this command in the Linux window:

```bash
$ awsstop
```

---

5 If the 3270 session displays a message `Unsupported Function`, simply use the 3270 Clear key to obtain the initial VTAM display. Some TN3270e emulators encounter this initial message and others do not.

6 This `shutdown` command is not a standard z/OS function; additional command names might be present. These trigger VTAMAPPL scripts to issue various commands involved in stopping z/OS.
This produces a number of messages. It might be necessary to press Enter to obtain the Linux prompt. Any 3270 windows may be closed at this point.

2.4.6 Startup messages

Messages such as the following are produced by the `awsstart` command:

```
AWSSTA014I Map file name specified: aprof13
0 Snapdump incident(s), RAS trace and RAS log files occupy 657046 bytes in /home/ibmsys1/z1090/logs
Associated files, logs, and core files occupy 10364 bytes in /home/ibmsys1/z1090/logs
```

Take a glance at these messages, because snapdump incidents are indications of an internal zPDT error, and if you want to work with IBM zPDT support, you will need this data. The number of bytes used for various logs and dumps is usually not significant unless it becomes too large. In general, zPDT manages these files automatically. However, if the numbers displayed become too large (many megabytes) and if you are not actively working on a problem with your zPDT support organization, you might want to clean up these files. This can be done by adding the `--clean` option the next time you issue an `awsstart` command:

```
$ awsstart aprof11 --clean
```

You can get the `--clean` behavior every time by setting a Linux shell environment variable Z1090_CLEAN=YES; however, we do not recommend doing this because it could easily result in the removal of important debugging information in the event of a zPDT failure.

2.4.7 Local volumes

The process for adding your own 3390 volumes is shown here:

- Allocate the volume using a zPDT utility:
  ```
  $ alcckd /z/WORK01 -d3390-3 (assuming you want a 3390-3 volume)
  ```

- Update the devmap to include the new volume. (Assume address AA0 for this example.)
  ```
  [manager]  
  name awsckd 0001
  ...
  device AA0 3390 3990 /z/WORK01
  ```

- Restart zPDT with the updated devmap.
- IPL z/OS with the new volume present. z/OS will detect an uninitialized volume and vary it offline.
- Create and run an ICKDSF job to initialize the volume:
  ```
  //BILLX JOB 1,OGDEN,MSGCLASS=X
  // EXEC PGM=ICKDSF,REGION=1M
  //SYSPRINT DD SYSOUT=*  /*
  //SYSPRINT DD SYSOUT=*  */
  INIT UNIT(AA0) NOVERIFY VOLID(WORK01) VTOC(0,1,14)
  */
  ```

- Vary the new volume online and begin using it:
  ```
  VARY AA0,ONLINE (on the z/OS console)
  ```
2.5 z/OS parameters

We recommend that you verify that the MIH values for both GRAF and COMM devices are set to 3 minutes in the IECIOSxx member in your PARMLIB. Three minutes is the default value, but we have seen installations with a lower value specified. For example, you could have a PARMLIB member named IECIOS00, as shown here:

```
MIH GRAF=3:00
MIH COMM=3:00
```

To use the IECIOS00 member, your IEASYSxx PARMLIB member should contain the following line:

```
IOS=00,
```

2.6 Multiple operating systems

We can install multiple System z operating systems, limited only by the disk space we have available. Every emulated 3390-3 volume uses approximately 2.8 GB of disk space. Many of our mobile computer disks were nominally 100 GB or larger. After allowances for Linux and other purposes we had at least 82 GB of usable space left. This translates to 29 3390-3 volumes. (There is no requirement to use 3390-3 volumes. A similar computation may be done for other or mixed 3390 sizes.)

We frequently placed emulated volumes on an inexpensive USB drive and we did not see a significant performance difference between this drive and the mobile computer internal hard disk. More recent mobile computer machines have even larger hard disks.

It is important to distinguish between installing additional emulated 3390 volumes (perhaps with a variety of operating systems), and using the volumes. We can, of course, only IPL a single system at any one time in a zPDT instance. The volumes that may be “seen” by that system depend on several factors:

- Does the current devmap contain all the desired volumes?
  - We can have multiple devmaps, each with a different selection of emulated volumes and assigned addresses, but we can have only one devmap specified when we start a zPDT instance. We cannot change the active devmap while zPDT is running.

- Do the device addresses in the devmap match suitable addresses in the IODF of the z/OS system?
  - For example, if one of the emulated 3390 volumes is assigned address 190 (in the devmap), then the default z/OS AD-CD IODF would not “see” the volume because this address is not in the IODF. (z/VM does not have predefined addresses for various device types, making this aspect of z/VM easier to use.)

- Duplicate disk volsers may not be present.
  - You may have duplicate volsers for emulated volumes on your PC disk, but the duplicates should not be present in a given devmap.

---

7 The zPDT developers have recommended this parameter be set. However, we seldom see the need for it and suggest that you might skip it unless you encounter 3270 connection timeouts.

8 This is the Missing Interrupt Handler function.

9 This statement ignores the possibility of running multiple z/OS guests under z/VM.

10 This is not completely true. We can change the volume mounted on an emulated disk drive or tape drive with the awsmount command.
It might not be possible to use the common addresses typically associated with an operating system.

For example, all the AD-CD documentation uses A80 as the IPL address for a z/OS AD-CD system. We can have two (or more) AD-CD systems represented in the devmap at the same time, but only one volume can have address A80 during any single execution of zPDT. This does not prevent us from IPLing any of the (multiple) AD-CD systems present, but we need to specify the correct address. An example might make this clearer:

<table>
<thead>
<tr>
<th>Address</th>
<th>VOLSER</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>A80</td>
<td>ZCRES1</td>
<td>IPL volume for z/OS 1.12 AD-CD system</td>
</tr>
<tr>
<td>A81</td>
<td>ZCRES2</td>
<td>Libraries for z/OS 1.12 AD-CD system</td>
</tr>
<tr>
<td>A82</td>
<td>ZCSYS1</td>
<td>Paging, spooling, VSAM for 1.12 AD-CD system</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A90</td>
<td>SBRES1</td>
<td>IPL volume for z/OS 1.11S system</td>
</tr>
<tr>
<td>A91</td>
<td>SBRES2</td>
<td>Libraries for z/OS 1.11S system</td>
</tr>
<tr>
<td>A92</td>
<td>SBSYS1</td>
<td>Paging, spooling, VSAM for 1.11S system</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Assuming our devmap is configured for these addresses, we can `ip1 A80 parm 0A82CS` to run the 1.12 system or we can `ip1 A90 parm 0A92CS` to run the 1.11S system. In either case, the running z/OS system can access all the volumes of both z/OS systems. This is convenient for migration purposes. The volumes are readdressed by simply changing the addresses in the devmap.

We can run multiple System z operating systems at the same time by starting multiple zPDT instances, but this requires more resources (especially PC memory). Details about using multiple instances are in Volume 3 of this documentation series.
LANs and TCP/IP

LAN setup can become complex and many variations are possible. We have selected five basic scenarios as possible starting points. We strongly suggest that your initial zPDT usage be with scenario 1, which has no System z TCP/IP functions.¹ The second scenario is then a simple migration from the first one. The third, fourth, and fifth scenarios provide different ways to connect zPDT functions to an external network. The key difference between these last three scenarios is whether you have an assigned, fixed IP address that can be used with your z/OS (or z/VM, or z/VSE, or Linux for System z).

This chapter does not address more complex LAN usage, such as might be used for multiple guests under z/VM. Several elements of z/VM usage are addressed by examples in the third book in this series, but there are many combinations and permutations that we cannot address. Again, we strongly suggest you start with the basic scenarios described in this chapter. After working through these you should then be familiar with the elements of LAN usage that are unique to zPDT.

We also strongly suggest that you take time to study this chapter before starting your LAN setup. We assume you have some familiarity with z/OS system programming tasks. The following discussions are in terms of z/OS, but most of the concepts also apply to z/VM and z/VSE.

**Important:** LAN setup is not part of the zPDT product. The examples in this chapter might help you decide how to configure your TCP/IP setup, but you must provide the networking skills to verify and implement your own design.

¹ If you follow the installation instructions in the first two chapters, you will be close to this environment.
3.1 Overview of LAN usage

There are four key factors that permeate this discussion for you to keep in mind:

- We have Linux TCP/IP and z/OS TCP/IP. These are two functions that operate separately. Always be aware of which TCP/IP is under discussion.
- You do not need any z/OS LAN functions (or z/OS TCP/IP functions) for 3270 access to z/OS. Access can be through the aws3274 device manager and appears as local, channel-attached terminals to z/OS. This is our first scenario.
- z/OS and the base Linux cannot communicate with each other through the same LAN adapter. Both can share the same hardware LAN adapter for all TCP/IP functions except communicating with each other. zPDT implements a tunnel pseudo-LAN to bypass this restriction.
- Standard z/OS is not a DHCP client. You cannot simply plug z/OS into any LAN outlet on your office wall. To connect to z/OS TCP/IP you must have a fixed IP address that is valid on your physical LAN segment.

At the time of writing, Linux bonding of several LAN adapters to create a single virtual adapter has not been tested with zPDT.

3.1.1 Three 3270 interfaces

There may be three 3270 interfaces with z/OS:

- The aws3274 device manager accepts TN3270e connections (from the local Linux host or over the Linux TCP/IP network.) The Linux TCP/IP port number for this connection is specified in the 3270port parameter in the devmap. z/OS sees these 3270 sessions as local, channel-attached, non-SNA, DFT terminals. Such terminals are suitable for z/OS operator consoles and VTAM use. z/OS TCP/IP is not involved and does not need to be running.
- z/OS TCP/IP provides TN3270e connections. Terminals connected this way are not usable as z/OS operator consoles. TN3270e connections through z/OS TCP/IP are routed to VTAM and may be used as TSO terminals, IBM CICS® terminals, and so forth. z/OS TCP/IP must be configured to use an OSA-Express adapter (in either non-QDIO or QDIO mode). The OSA-Express functions are emulated by the awsosa device manager.
- z/OS VTAM potentially could work with SNA 3270 Ethernet connections, working through the awsosa device manager. However, SNA operation with zPDT is not supported.

The same Ethernet adapter can be used for Linux functions, such as telnet, aws3274, ftp, and so forth, and also for OS/390 connections. Important concepts include the following:

- An emulated OSA-Express interface requires a hardware Ethernet adapter port on the underlying Linux system (or a tunnel interface, as described later). A mobile computer normally has one integrated Ethernet port. (It may also have integrated wireless functions, which count as an additional port.) Additional Ethernet ports may be added by using PC (PCMCIA-type) cards, although few zPDT users are expected to need more than one Ethernet adapter.

---

2 In strict Linux terminology we do not have a tunnel interface; we use a tap interface rather than a tun interface. We use the word tunnel in a more generic sense.

3 A TN3270 connection (as opposed to a TN3270e connection) will be accepted, but extended data stream capabilities are not present and some z/OS functions might not work correctly.

4 We describe this as an OSA-Express2 device manager, but this description is only approximate. This device manager has attributes of the original OSA, OSA-Express, and OSA-Express2 channels available on larger System z machines.
An emulated OSA-Express interface operating in QDIO mode is used only for z/OS TCP/IP (or z/VM TCP/IP, and so forth). QDIO mode is also known as OSD mode.

An emulated OSA-Express interface operating in non-QDIO mode can be used by z/OS TCP/IP or SNA (although SNA is not supported at this time). Non-QDIO mode is sometimes known as LCS mode or OSE mode.

If you want to communicate between Linux TCP/IP and OSA TCP/IP on the same PC, a tunnel environment must be established.\(^5\)

### 3.2 Basic QDIO setup

The following examples assume that OSA is used as a QDIO device, as opposed to an LCS (non-QDIO) device. An overview of LCS usage is provided in the third volume in this documentation series. We recommend using QDIO operation. For QDIO operation:

- Three OSA devices are needed for a TCP/IP connection. The first should be at an even-number address.
- z/OS devices involved must be defined as OSA devices in the z/OS IODF.
- A TRLE definition is needed in VTAMLST, pointed to by ATCCON00 in VTAMLST.
- The z/OS TCPIP PROFILE uses a IPAQENET device type.

The awsosa definitions must include path numbers and path types for OSA devices. The path type is OSD (for QDIO). The path is determined with the `find_io` command on your system. We cannot predict exactly what that path might be.

Recent z/OS AD-CD systems include OSA devices starting at device number 400. When using the QDIO interface to the emulated OSA-Express2 function, the key parameters might look like the following.\(^6\)

**Devmap**

```plaintext
[manager]
name awsosa 22 --path=F0 --pathtype=OSD
device 400 osa osa
device 401 osa osa
device 402 osa osa
```

Or the following for a tunnel connection

```plaintext
[manager]
name awsosa 22 --path=A0 --pathtype=OSD --tunnel_intf=y
device 400 osa osa
device 401 osa osa
device 402 osa osa
```

**z/OS VTAMLST**

```plaintext
OSATRL1 VBUILD TYPE=TRL
OSATRL1E TRLE LNCTL=MPC,READ=(400),WRITE=(401),DATAPATH=(402), X PORTNAME=PORTA,MPLEVEL=QDIO
```

---

\(^5\) Another method would use two Ethernet adapters connected to the same network, one for the base Linux and one for z/OS. We do not recommend this method.

\(^6\) This example might be appropriate for an OSA LAN connection through an Ethernet adapter.
**z/OS TCP/IP Profile**

```
DEVICE PORTA MPCIPA
LINK ETH1 IPAQENET PORTA
HOME 192.168.0.61 ETH1
...
BEGINRoutes
  ;     Destination  Subnet Mask    FirstHop    Link  Size
ROUTE 192.168.0.0 255.255.255.0 = ETH1 MTU 1492
ROUTE DEFAULT 192.168.0.1 ETH1 MTU 1492
ENDRoutes
...
START PORTA
```

The presence of the `--tunnel_intf` parameter in the devmap indicates that a tunnel (tap device) connection will be created. The default address for the Linux side of a tunnel is 10.1.1.1. The `--path` value is the CHPID number returned by the `find_io` command. The CHPID value is usually A0 for a tunnel connection, F0 for a direct Ethernet LAN connection, or F8 for a wireless connection, but these values should be verified with `find_io`.

---

**Important:** zPDT Version 1 Release 3 (“GA3”) has changed the data displayed by the `find_io` command, and new specifications exist in the awsosa device statement to handle a variety of NIC names used by newer Linux distributions. Read Chapter 10 of the third book in this series (SG24-7723-04 or later) for details. The examples in this chapter use path names F0, F8, and A0, but these might not always be appropriate. Path names used in devmaps from previous zPDT releases might not be the same with this release.

---

A VTAM major node known as a TRL is required in VTAMLST for QDIO operation. This VTAM node must be active before TCP/IP can be started. The VTAMLST ATCCON00 member must point to the TRL entry in VTAMLST.

The MIH value set for the write interface (401 in the example) should be set to at least 30 seconds.\(^7\) (However, we find that the default value is usually sufficient.)

The PORTNAME (in the TRLE), the DEVICE name (second field), the LINK parameter (fourth field), and the START name must match. The name is arbitrary, but it must be the same in all four places.

### 3.3 Five scenarios

Five scenarios are described in this chapter. We use z/OS as the target operating system in these descriptions, but z/VSE or z/VM or Linux for System z could be used with appropriate adjustments. We use QDIO (OSD) interfaces for these examples, although LCS (OSE) TCP/IP interfaces could be used.\(^8\) QDIO usage in z/OS requires parameters in VTAMLST, and these are included in the setup examples.

---

\(^7\) This was recommended by the developers. In routine usage we have not seen any problems using the default z/OS MIH values.

\(^8\) We recommend using QDIO interfaces for several reasons. QDIO connections are faster and are not as exposed to packet saturation as LCS interfaces.
The five scenarios are listed here:

1. No TCP/IP function is used in z/OS. Only emulated local 3270 connections are used between the base Linux and z/OS. The base Linux could be connected to a larger LAN; this is transparent to z/OS. Emulated 3270 sessions from the base Linux or from the LAN can connect to z/OS, where they appear as local, channel-attached 3270 sessions.

2. z/OS TCP/IP is used to connect to the base Linux via a tunnel function. All z/OS TCP/IP activity is directed to the tunnel. This allows TCP/IP connections between the base Linux and z/OS, and these might be used for ftp, telnet (to UNIX System Services), TN3270 connections directly to z/OS TCP/IP, and so forth. The base Linux could be connected to an external LAN, but this is transparent to z/OS and external LAN connections cannot be made to z/OS.

3. The same basic setup as scenario 2, but with additional customization to enable a simple NAT function in the base Linux. This permits TCP/IP connections from z/OS to the external LAN, but not from the external LAN to z/OS. (That is, only outgoing TCP/IP sessions may be initiated. With additional NAT/iptables customization, incoming TCP/IP connections from the external LAN to z/OS could be handled. This additional customization might involve non-standard port numbers for either Linux or z/OS.)

4. Instead of the NAT functions used in option 3, an additional OSA interface is used by z/OS to connect to the LAN. A fixed IP address is needed for z/OS. TCP/IP communication between z/OS and the base Linux is through the tunnel.

5. A different NAT function is used that allows incoming and outgoing connections to z/OS. In this scenario, only the tunnel OSA is used by z/OS and both tunnel and external LAN traffic flow through it. The z/OS setup is the same as for scenario three, but the base Linux setup is different. A fixed IP address is needed for z/OS.

In these examples, the names assigned to the OSA interfaces for z/OS are eth1 and eth2 (if needed). These examples use eth1 for the tunnel connection to the base Linux. These names are completely arbitrary.

OSA definitions for zPDT require the use of a CHPID number for the path parameter. The CHPID path for the tunnel is assumed to be A0 and the path for the external LAN is assumed to be F0. These paths should be verified with the zPDT find_io command. This command might not display information for tap devices until after zPDT has been started at least once with a tunnel definition included in the devmap.

Figure 3-1 suggests a way to select the most appropriate scenario.

---

9 The correct terminology is “connect via a Linux tap interface.” However, we use the term “tunnel” in a generic sense to describe this connection.
Scenarios 3 and 5 produce similar results in different ways. Scenario 3 requires more customization in z/OS TCP/IP and scenario 5 requires more customization in the base Linux.

### 3.3.1 Scenario 1

Scenario 1 is illustrated in Figure 3-2.

With this option, no z/OS TCP/IP setup is required and z/OS TCP/IP does not need to be active. (The AD-CD z/OS system starts TCP/IP by default; you could remove the associated start statement in a PARMLIB member.) You can use up to 31 TN3270 sessions for
connections to TSO or other VTAM functions. (One TN3270 session is needed for the IBM MVS™ console.) A variety of TN3270 emulators can be used, including x3270 and PCOMM. These 3270 emulator sessions might be in the base Linux or through a LAN connection to the base Linux, if this exists. (The LAN connection to an external network is optional.) The only upload/download method between the base Linux and z/OS is by using the IND$FILE functions.\textsuperscript{10}

No OSA definitions are needed in the devmap. The relevant devmap definitions are for the 3270 port and for several local 3270 devices.

```
[system]
....
3270port 3270 #the port number is arbitrary. 3270 is easy to remember.

[manager]
name aws3274 0001
device 0700 3279 3274 #Address 0700 is the MVS console in the AD-CD systems
device 0701 3279 3274 #Other systems may want different addresses
device 0702 3270 3274
...
```

Based on this example, connections from the base Linux might start as follows:

```
$ x3270 -port 3270 localhost &
```

A connection from the external LAN might be started as follows:

```
$ x3270 -port 3270 9.111.222.123 &
```

This is assuming the DHCP address assigned to the local Linux is 9.111.222.123. You can find the assigned DHCP address for your Linux with the `/sbin/ifconfig` command.

Making a TN3270 connection to aws3274 on the base Linux (or any other service on the base Linux) from an external LAN, might present routing difficulties. The LAN must have route definitions that allow both the external TN3270 system and the base Linux to find routes to each other. This routing requirement is not unique to zPDT. If you have a firewall running in the base Linux, you might need to create a “hole” in it for the connection to port 3270 (or whatever port you defined for aws3274 connections). If your firewall is based on iptables (as is common for most current Linux releases), commands such as the following might be used:

```
$ su (switch to root)
# iptables -I INPUT -p tcp --dport 3270 -j ACCEPT
# exit (leave root)
```

These commands would be entered through a Linux terminal window. In general, details about managing your Linux firewall and your external routing controls is beyond the scope of this document.

### 3.3.2 Scenario 2

The second scenario builds on the first one and adds a direct TCP/IP connection between z/OS and the base Linux, as shown in Figure 3-3.

\textsuperscript{10} These are often known as file transfer functions in the 3270 emulators.
This TCP/IP connection is through a “tunnel” interface between z/OS and the base Linux. The physical LAN adapter is not involved. The 10.x.x.x IP addresses shown are arbitrary, but we recommend using non-routable addresses on an isolated subnet. The tap interface (and associated IP address) are created automatically when zPDT is first started (assuming the correct OSA definitions are in the devmap). No additional Linux setup is needed. z/OS TCP/IP must include an OSA definition for its interface.

Recent AD-CD systems include OSA devices starting at device number 400. When using the QDIO interface to the emulated OSA-Express2 function, the key parameters might look like the following example:

Devmap

(The 3270 port and aws3274 device manager definitions used in the previous example should be included here.)

[manager]
name awsosa 22 --path=A0 --pathtype=OSD --tunnel_intf=y
device 400 osa osa
device 401 osa osa
device 402 osa osa

z/OS VTAMLST TRL definition

OSATRL1 VBUILD TYPE=TRL
OSATRL2E TRLE LNCTL=MPC,READ=(400),WRITE=(401),DATAPATH=(402), X
PORTNAME=PORTA,MPCLEVEL=QDIO

z/OS TCP/IP Profile

... 
DEVICE PORTA MPCIPA
LINK ETH1 IPAQENET PORTA
HOME 10.1.1.2 ETH1 
...
BEGINRoutes
; Destination Subnet Mask FirstHop Link Size
ROUTE 10.0.0.0 255.0.0.0 = ETH1 MTU 1492

11 This connection cannot use a physical LAN adapter on current Linux systems.
The external LAN connected to Linux and the “tunnel LAN” between Linux and z/OS are completely separate in this example, and there is no communication between them. There is no connection from z/OS to the outside world, but all normal TCP/IP functions between the base Linux and z/OS may be used. Examples (from the Linux side) include:

```bash
$ x3270 -port 3270 localhost & (connect via “local 3270” channel)
$ x3270 10.1.1.2 & (connect via z/OS TCP/IP)
$ ftp 10.1.1.2 (connect to z/OS FTP)
$ telnet 10.1.1.2 1023 (connect to z/OS UNIX System Services)
```

From the z/OS TSO side, we might use a command such as this one to connect to ftp on the base Linux. (This assumes you have ftp installed and available on the base Linux.)

```bash
ftp 10.1.1.1 (entered in ISPF option 6, for example)
```

### Tunnel IP addresses

The IP addresses used for the tunnel (10.1.1.1 and 10.1.1.2 in the examples) are not related to any other IP addresses you might use. They are not related to any external IP addresses. They should not be on the same subnet as any external IP addresses you might use. These tunnel addresses are solely for use between the base Linux and TCP/IP stacks running within the zPDT environment.

The IP address for the base Linux side of the tunnel defaults to 10.1.1.1 (for the first tunnel OSA), but may be changed in the devmap. The address at the other end (z/OS or z/VM) must be different but should be on the same subnet as determined by the netmask.

The 10.x.x.x addresses (and 192.168.x.x addresses) are not routable. You should not attempt to make them visible to your network users.

In our examples, the 192.168.x.x addresses are assumed to be on the “local side” of a router, which is probably a NAT router. As used in our examples, the 192.168.x.x addresses are visible and usable by other systems connected to the local side of this router.

### 3.3.3 Scenario 3

Scenario 3 is depicted in Figure 3-4.

---

\[12\] The AD-CD z/OS system uses port 1023 for a simple telnet connection to UNIX System Services.
We can take the scenario 2 setup and extend it to connect z/OS to the external LAN by using a NAT\textsuperscript{13} function in the base Linux. This requires a more complex setup. However, it has the major benefit that an assigned, fixed IP address is not needed for z/OS. With this setup, z/OS has the fixed address 10.1.1.2 (in our examples), but this is not an externally assigned address; it is visible only internally in our local Linux system.

The following text describes how to do this dynamically (through commands each time the system is started). This example is based on openSUSE 11.0; there might be minor differences for other Linux distributions.

The first step is to create a Linux file in the zPDT home directory, as shown here. We named this file `masq`.

```
$ cd ~
$ touch masq
$ gedit masq  
```

```
(the following lines start in column 1 in the file)
```
```
if [[ $EUID -ne 0 ]]; then
  echo 'You must su to root to run this command' >&2
  exit 1
fi
```
```
echo 'Your firewall must be enabled for this command to be meaningful'
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -F FORWARD
iptables -P FORWARD ACCEPT
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
iptables -I INPUT -p tcp --dport 3270 -j ACCEPT
echo 'Done. Exit from root'
```

Use the same devmap and z/OS PROFILE parameters shown for scenario 2. Assuming your base Linux is connected to an external LAN (either with a fixed IP address or a DHCP address), activate your Linux firewall (if not already done) and activate the iptable changes:

```
$ cd ~
$ su
`switch to root`
# ./masq  
`execute the command script we just created`
# exit  
`exit from root`
```

\textsuperscript{13} NAT is Network Address Translation.
You should be able to access external network sites from z/OS. For example, at the time of writing the www.redbooks.ibm.com site is IP address 170.225.15.45. From a TSO command line you can try the following command:

```
ftp 170.225.15.45
```

You need an appropriate userid and password to use this site, but you should see the ftp connection. You should be able to ping any Internet sites that are known to respond to pings,\(^{14}\) if you can find their numeric IP address.

You might need to use the passive option with ftp connections. Complex applications that, once initiated from z/OS, might trigger incoming connections on other ports might not work. Incoming connections to port 3270 on the base Linux are allowed by our script; this provides a *hole* in the standard Linux firewall for using the local 3270 connections.

**Incoming connections**

This setup has complications for incoming (from the external LAN) connections. Remember that external systems see only the IP address of the base Linux. (This is probably a DHCP-assigned address, and must be communicated to external users in some manner.) If an external user attempts to connect to port 23, for example, does the user want Linux port 23 or z/OS port 23? (This assumes the user can get through the Linux firewall, which is another complication.) Port 23 is a well-known (default) port number for telnet connections (including TN3270e telnet).\(^{15}\)

One way around this problem is to use a non-standard port number for telnet on either Linux or z/OS. Another way around the problem is to simply disallow port 23 connections to either Linux or z/OS. (The issue applies to all port numbers; we are using port 23 as an example.)

As an example, adding the following line to the script would route external connections for port 23 to z/OS:

```
iptables -t nat -A PREROUTING -p tcp -i eth0 --dport 23 -j DNAT --to 10.1.1.2
```

If we add this command to our script, then an external user would have two paths for a TN3270e connection (assuming the Linux IP address is 192.168.0.2):

```
$ x3270 192.168.0.2 & Connect directly to z/OS TCP/IP port 23
$ x3270 192.168.0.2:3270 & Connect to Linux port 3270 (aws3274)
```

Be aware that after this `iptables` command is issued, we no longer have a way to connect to Linux port 23.

Extending this example to other ports, and determining what services might be wanted on both Linux and z/OS, becomes more complex and depends on the exact base Linux configuration for firewalls and available services.

### 3.3.4 Scenario 4

This setup provides a direct connection from z/OS to the external LAN. A NAT function is not used. Only a single physical LAN adapter is needed and is used by both Linux and z/OS. z/OS must have an external assigned, fixed IP address for this to work. Our example uses address 192.168.0.61, but this is not a real assigned IP address\(^{16}\). You must have a proper assigned IP address for this option to work. Remember that assigned, fixed IP addresses are

---

\(^{14}\) It appears that most common Internet sites no longer respond to pings. You can verify your results by issuing pings from the base Linux system.

\(^{15}\) Some Linux systems have completely dropped telnet service (that listens on port 23); our comments apply to all ports.
not that portable; they must be used on a physical LAN segment that is the router target for
the associated subnet.

Figure 3-5 illustrates this configuration. The figure shows two logical connections to the
external network, but this is accomplished by a single physical cable connection.

With this configuration the IP functions of z/OS and the base Linux are quite separate. The
tunnel addresses (10.1.1.x) are not visible from the external network.

The various definition files should contain the following:

Devmap

(The 3270 port and aws3274 device manager definitions used in the
previous examples should be included here.)

[manager]
name awsosa 0024 --path=A0 --pathtype=OSD --tunnel_intf=y
device 400 osa osa
device 401 osa osa
device 402 osa osa

[manager]
name awsosa 0022 --path=F0 --pathtype=OSD
device 404 osa osa
device 405 osa osa
device 406 osa osa

z/OS VTAMLST

OSATRL1 VBUILD TYPE=TRL
OSATRL1E TRLE LNCTL=MPC,READ=(400),WRITE=(401),DATAPATH=(402),
PORTNAME=PORTA,MP LEVEL=QDIO
OSATRL2E TRLE LNCTL=MPC,READ=(404),WRITE=(405),DATAPATH=(406),

16 IP addresses 192.168.0.0 are “nonroutable” addresses that can be used on a local network that is not connected
to external networks.
PORTNAME=PORTB, MPCLEVEL=QDIO

z/OS TCP/IP Profile

DEVICE PORTA MPCIPA
LINK ETH1 IPAQENET PORTA
HOME 10.1.1.2 ETH1

DEVICE PORTB MPCIPA
LINK ETH2 IPAQENET PORTB
HOME 192.168.0.61 ETH2

...BEGINRoutes
; Destination Subnet Mask FirstHop Link Size
ROUTE 10.0.0.0 255.0.0.0 = ETH1 MTU 1492
ROUTE 192.168.0.0 255.255.255.0 = ETH2 MTU 1492
ROUTE DEFAULT 192.168.0.1 ETH2 MTU DEFAULTSIZE
ENDRoutes
...
START PORTA
START PORTB

Again, remember that the 192.168.x.x addresses cannot be used for “real” Internet connections. You must supply your assigned, fixed IP address and also supply a default address for your network connection.

With this scenario, connections to and from z/OS and the external network are independent from base Linux connections. However, you must still use the 10.1.1.x addresses for TCP/IP communication between the base Linux and z/OS. That is why we show two OSA definitions and connections in this example.

3.3.5 Scenario 5

Figure 3-6 Scenario 5 connectivity
We can take the scenario 2 setup and extend it to connect z/OS to the external LAN by using a NAT function in the base Linux in a different way. This method requires an assigned, fixed IP address for z/OS.

A single OSA interface in z/OS handles both tunnel traffic (to the base Linux) and external IP traffic. Incoming connections to z/OS are handled, as well as outgoing connections.

The following text describes how to do this dynamically (using commands each time the system is started). Note that this method uses an IP alias address in the base Linux.

The first step is to create a Linux file in the zPDT home directory. We use a more elaborate script file here to better allow it to be expanded in the future. We named this file `nat2`

```
$ cd ~
$ touch nat2
$ gedit nat2 (the following lines start in column 1 in the file)
if [[ $EUID -ne 0 ]]; then
  echo 'You must be root to run this command' 1>&2
  exit 1
fi
echo 'Your firewall must be enabled for this command to be meaningful'
CHPID_A0_INTERFACE=eth0
CHPID_A0_EXTERNAL_IP=192.168.0.70 (your assigned IP address)
CHPID_A0_EXTERNAL_BC=192.168.0.255 (broadcast address for it)
CHPID_A0_EXTERNAL_NM=255.255.255.0 (net mask for it)
CHPID_A0_VIRTUAL_IP=10.1.1.2
echo 1 > /proc/sys/net/ipv4/ip_forward
echo 'IP forwarding set'
iptables -t nat -F
echo 'nat table flushed'
echo 'External IP address for System z is ' $CHPID_A0_EXTERNAL_IP
echo 'Real LAN interface is ' $CHPID_A0_INTERFACE
echo 'Tap (tunnel) address for System z is ' $CHPID_A0_VIRTUAL_IP
echo 'External netmask and broadcast address are ' $CHPID_A0_EXTERNAL_NM \ $CHPID_A0_EXTERNAL_BC
ifconfig $CHPID_A0_INTERFACE:0 $CHPID_A0_EXTERNAL_IP netmask \ $CHPID_A0_EXTERNAL_NM broadcast $CHPID_A0_EXTERNAL_BC up
iptables -t nat -A POSTROUTING -o $CHPID_A0_INTERFACE -s \ $CHPID_A0_VIRTUAL_IP/32 -j SNAT --to $CHPID_A0_EXTERNAL_IP
iptables -t nat -A PREROUTING -i $CHPID_A0_INTERFACE -d \ $CHPID_A0_EXTERNAL_IP/32 -j DNAT --to $CHPID_A0_VIRTUAL_IP
echo 'Done. Please exit from root'
```

Use the same devmap and z/OS PROFILE parameters shown for scenario 2. Assuming your base Linux is connected to an external LAN (either with a fixed IP address or a DHCP address), activate your Linux firewall (if not already done) and activate the iptable changes:

```
$ cd ~
$ su (switch to root)
```

Footnote:
17 Four lines in this file end with a back slash (\) to indicate that the logical line is continued on the next printed line. You can enter each of these lines as a single long line (without the back slash).
You should be able to access external sites from z/OS. External LAN users can connect to your base Linux by using its DHCP address\(^\text{18}\) and connect to z/OS by using its assigned fixed address.

If you use LCS mode connections (non-QDIO, OSE) for some reason, this scenario has an additional benefit. Unwanted packets are filtered out at the base Linux level instead of being forwarded to z/OS, where many cycles might be required to filter out unwanted packets.

### 3.3.6 z/OS resolver

**Tip:** The later z/OS 1.13 AD-CD systems already have a default resolver configuration that is slightly different than what is described here.

The z/OS resolver can access an external name server using the connections in scenarios 3, 4, and 5. There are a number of ways to do this, and the following is one example. It involves creating or changing several data sets (or members) in z/OS. Our example uses data set names from the AD-CD z/OS 1.12 winter release; you might need to adjust these names for your z/OS system.

```plaintext
[create member ADCD.Z112.TCPPARMS(RSSETUP)]
GLOBALTCPIPDATA('ADCD.Z112.TCPPARMS(GLOBAL)')
GLOBALIPNODES('ADCD.Z112.TCPPARMS(IPNODES)')
COMMONSEARCH

[create member ADCD.Z112.TCPPARMS(GLOBAL)]
TCPIPJOBNAME TCPIP
HOSTNAME yourname
DOMAINORIGIN ADCD.ITSO.IBM.COM (use an appropriate name here)
DATASETPREFIX TCPIP
MESSAGECASE MIXED
NSINTERADDR 167.206.251.130 (specify an appropriate name server)
NSPORTADDR 53
RESOLVEVIA UDP
RESOLVERTIMEOUT 10
RESOLVERUDPRETRIES 1
LOOKUP DNS

[create member ADCD.Z112.TCPPARMS(IPNODES)] (may not be needed)
167.206.251.130 nameserver
```

You need a nameserver that is available to you. You might find one by issuing a Linux command such as `nslookup www.ibm.com` and noting what server was used to resolve the name.

The AD-CD system (at the time of writing) starts the resolver using the default name RESOLVER. Assuming this situation, issue the following commands at the MVS console:

```plaintext
F RESOLVER,DISPLAY (verify that the resolver is running)
F RESOLVER,REFRESH,SETUP=ADCD.Z112.TCPPARMS(RSSETUP)
```

\(^{18}\) You could also have an assigned, fixed address for your base Linux.
where the specified setup file is the one created earlier. When this is working correctly, you should be able to use the following commands from TSO:

- NSLOOKUP WWW.REDBOOKS.IBM.COM
- PING WWW.REDBOOKS.IBM.COM
- FTP WWW.REDBOOKS.IBM.COM

The www.redbooks.ibm.com site is just an example.

The use of an Internet name server can be more complex than indicated here. If you fully connect to a name server you should take care to specify an appropriate domain and host name for your own system that does not duplicate any existing name that might be resolved by an Internet name server.

### Permanent changes

The resolver changes can be made permanent (automatically used by z/OS TCP/IP) by changing the BPXPRMxxx members in your PARMLIB. To do this, first create the following procedure in SYS1.PROCLIB(RESOLVAD):\(^\text{19}\)

```
//RESOLVAD PROC PARMS='CTRACE(CTIRES00)'
//EXBREINI EXEC PGM=EZBREINI,REGION=0M,TIME=1440,PARM=&PARMS
//SETUP DD DSN=ADCD.Z112.TCPPARMS(RSSETUP),DISP=SHR,FREE=CLOSE
```

Next, identify the BPXPRMxxx member or members in ADCD.Z112.PARMLIB that are used by your IPL parameters. This is likely to include BPXPRMCS for recent AD-CD releases. You can change all the BPXPRMxxx members that you might use. Edit these members and add the following line as the last line in each member (after a blank line):

```
RESOLVER_PROC(RESOLVAD)
```

The ADCD.Z112.PARMLIB name should be adjusted to match your AD-CD release. The member names RESOLVAD, RSSETUP, GLOBAL, and IPNAMES are arbitrary names. You will probably want to add the following command to your SHUTDOWN scripts\(^\text{20}\) to terminate the resolver:

```
P RESOLVAD
```

Making any iptables commands permanent varies with different Linux distributions. You might simply execute a small iptables script manually before starting zPDT. Ways to make the \textit{iptables} commands automatic (in the base Linux) vary with different Linux levels and distributions. An example of executing your \textit{iptables} commands automatically is shown here (for openSUSE 11.4):\(^\text{21}\)

- Decide on your \textit{iptables} commands.
- Edit file /etc/sysconfig/scripts/SuSEfirewall2-custom.
  - Place your commands in the section named \texttt{fw\_custom\_after\_chain\_creation()}.
    - Place your commands within the braces in this section.
- Edit file /etc/sysconfig/SuSEfirewall2
  - Locate the line with \texttt{FW\_CUSTOMRULES=""}, which is several hundred lines into the file.
  - Comment out the indicated line.
  - Uncomment the line that reads:
    - \texttt{FW\_CUSTOMRULES="/etc/sysconfig/scripts/SuSEfirewall2-custom"}

\(^\text{19}\) Member name RESOLVAD is arbitrary. The procedure should be in SYS1.PROCLIB because it is started under MSTR, the master scheduler.

\(^\text{20}\) These scripts are in the AD-CD PARMLIB data set, with names such as SHUTDOWN, SHUTALL, and so forth.

\(^\text{21}\) Thanks to Lennie Dymoke-Bradshaw for this tip.
With these changes, the openSUSE 11.4 system automatically executed the iptables commands during Linux startup.

See the following document for more resolver setup information: *Communications Server for z/OS V1R9 TCP/IP Implementation Volume1: Base Functions, Connectivity, and Routing*, SG24-7532.

### 3.3.7 Debugging LAN setups

Creating a working LAN environment can be frustrating because many details must work together. Sometimes the problem lies outside your environment, with external routers that are not configured for the “new” systems you are placing on the LAN. We suggest using a small, inexpensive personal router for initial zPDT LAN setup, as shown in Figure 3-7.

![Figure 3-7  LAN debugging setup](image)

The advantage of using this test setup is that there are no unknown external LAN complications. The “local connections” to the router can be served DHCP addresses by the router, or they can be assigned fixed addresses on the router subnet. These routers typically use IP addresses of 192.168.0.x or 192.168.1.x. Many such routers also provide wireless connections and you can also explore these. (We suggest initial use with a wired connection to Linux, just to simplify the first setup.)

After you have an environment like this working, you can then transfer operation to a larger network. Although this suggested environment is almost trivial, we have often found it quite useful.

Remember that the Linux firewall (if enabled) might affect any external connections. For initial debugging (especially in a private environment, as shown here) we suggest that the firewall be disabled until you verify that your basic LAN setup is working. We have observed that the Security Enhanced Linux (SEL) protection might need to be disabled or modified for some functions, such as ftp.
3.3.8 Performance problems

At the time of writing we were aware of two particular problems that impact OSA performance.

- If frames larger than expected are used, there might be an excessive number of frames dropped (causing a retransmission). This might not be noticed unless careful measurements or comparisons are made. We believe this problem is resolved by including the systcl parameter that is now recommended in the first chapter of this document:

  net.core.rmem_max=1048576

- If advanced Linux kernels are installed, there might be a drastic slowdown of OSA performance that would be immediately obvious. This is due to Linux attempting to offload checksum functions into the adapter, which is not acceptable to the current awsOSA implementation. One solution is to use a Linux command:

  # ethtool -K eth0 rx off
  # ethtool -K em1 rx off  
  (newer style of NIC naming)

Unfortunately, this command must be entered after each Linux boot. The problem was first noticed with a build of Linux kernel 2.6.36.2; the ethtool must be at least at level 2.6.33.

IBM has not published performance specifications for OSA. Informal observation indicates that ftp throughput may be in the 5 - 8 megabytes/second range, assuming an unconstrained network in a dedicated environment. If your performance is much worse than this, the two problems mentioned here might be reviewed.

3.3.9 Detailed scenario comparison

Table 3-1 provides a summary of the characteristics of these five scenarios. Note the from and to words in the descriptions.

<table>
<thead>
<tr>
<th>Scenario characteristics</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of local 3270 sessions from base Linux (including external LAN) to z/OS (using the aws3274 manager)</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>Number of OSA definitions in z/OS</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Number of &quot;command files&quot; needed to run in base Linux (iptables)</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>External LAN connection from/to z/OS (not counting local 3270 sessions)</td>
<td>no</td>
<td>no</td>
<td>yes&lt;sup&gt;b&lt;/sup&gt;</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>TN3270, ftp, telnet from local Linux to z/OS TCP/IP, ftp from z/OS to local Linux</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>TN3270e from external LAN to z/OS</td>
<td>no</td>
<td>no</td>
<td>maybe&lt;sup&gt;c&lt;/sup&gt;</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>ftp from z/OS to external LAN</td>
<td>no</td>
<td>no</td>
<td>yes&lt;sup&gt;d&lt;/sup&gt;</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Externally assigned IP address needed for z/OS</td>
<td>no</td>
<td>no&lt;sup&gt;e&lt;/sup&gt;</td>
<td>no&lt;sup&gt;e&lt;/sup&gt;</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Telnet connection to UNIX System Services</td>
<td>no</td>
<td>yes, only from base Linux</td>
<td>yes, only from base Linux</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
### 3.4 Wireless connections

Wireless connections can be used by Linux TCP/IP or by OSA. Consider the following details:

- **Linux typically sees a wireless connection as device ath0, wlan0, or eth0. The `find_io` command lists a wireless interface along with Ethernet interfaces and associates a CHPID with it. (The CHPID address for a wireless adapter is normally F8.) You can then use this CHPID number as the path parameter for defining an awsosa interface.**

- We cannot provide a cookbook for activating your wireless link for Linux, but you need to have stable Linux wireless operation before trying to extend it to zPDT usage.

- We have noticed that the more recent Linux distributions provide much more convenient wireless setup than earlier Linux distributions.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Browser connection <em>from</em> base Linux to z/OS</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Browser connection <em>from</em> external LAN to z/OS</td>
<td>no</td>
<td>no</td>
<td>maybe&lt;sup&gt;c&lt;/sup&gt;</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Browser or ftp connection <em>from</em> external LAN to base Linux</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

- The local 3270 sessions are based on LAN connections to Linux (to the aws3270 device manager). The LAN connection is not to z/OS.
- Only outgoing connections from z/OS to the external LAN may be used unless additional `iptables` commands are used. The additional command functions are likely to require the use of non-standard port numbers.
- This would be an incoming connection and is not possible in the most basic `iptables` example. Incoming connections may be accepted when an additional `iptables` setup is used.
- This is an outgoing connection and may be used. A passive ftp connection may be needed.
- The "locally fixed" IP address (10.1.1.2 in the examples) is not an assigned IP address.

The fundamental difference between scenario 3 and scenarios 4 and 5 is that the latter require an assigned, fixed address for z/OS. Scenario 3 has complications for incoming connections (from the external LAN) to z/OS; 4 and 5 do not have this restriction. The primary difference between 4 and 5 is where the external LAN interface appears; with scenario 4 it is defined to z/OS; with scenario 5, it is defined in the base Linux.

### 3.5 Telnet to z/OS

If you elect to install the tunnel connection as described earlier (and as shown in the complete example listed in the Appendix), you can connect from the base Linux to z/OS by both telnet (in line mode) or by a TN3270e client such as x3270. Using the IP addresses from our examples, the Linux commands would be as shown here:

```bash
$ x3270 10.1.1.2 &  (to start a TN3270 session via z/OS TCP/IP)
$ telnet 10.1.1.2 1023  (line-mode telnet session via z/OS TCP/IP)
```

The 1023 parameter in the `telnet` command specifies the port number that the AD-CD TCP/IP and UNIX System Services profile use for Telnet connections. This port number (1023) is not standard, and probably applies only to the AD-CD z/OS system.
3.6 Choices

Which 3270 connection mode is better? If only simple 3270 connections are needed (and not more than 32 sessions are needed), then the use of basic aws3274 connections (shown in scenario 1) is better. This is simpler to set up and does not require OSA or z/OS TCP/IP to be configured or started.

Which CHPID mode should you use for OSA connectivity? QDIO mode has many benefits for TCP/IP usage on a larger System z; it reduces the System z workload and provides automatic sharing of the adapter across multiple LPARs. These considerations do not fully apply to a zPDT system. The following points are relevant:

- QDIO operation offloads some processing from the zPDT CP to the Linux processor. The offloading is not as much as on a larger machine, but it helps. It also reduces the number of System z instructions needed to maintain LAN I/O operation. In informal operation we noticed that ftp performance was about 20% faster with QDIO than with LCS.
- QDIO operation is only for TCP/IP; it does not handle SNA.
- QDIO can provide VSWITCH, IPv6, and Enterprise Extension connections.
- QDIO operation requires that OSA devices be defined in z/OS. These were not present in AD-CD systems prior to z/OS 1.8. The devices could be added to earlier AD-CD systems, using HCD to create a new IODF, or might be present in z/OS systems not derived from the AD-CD packages.
- Non-QDIO operation can mix TCP/IP and SNA (or handle just SNA or just TCP/IP). However, SNA operation with zPDT is not supported at this time.
- Suitable non-QDIO (LCS) devices have been defined in all the earlier z/OS AD-CD systems. (These are the CTCs starting at address E20.)
- The required OAT table is automatically updated when QDIO is used. The default OAT table is probably satisfactory for non-QDIO TCP/IP usage and might be satisfactory for SNA usage (although SNA is not supported at this time). The OSA/SF utility functions are used (if needed) to manipulate the OATs.

Other than these points, there is no practical difference between using QDIO or non-QDIO on a zPDT system. In particular, the user at a TN3270 TSO session cannot detect the difference. Normal TCP/IP functions, such as ftp and Telnet, do not detect any differences. If you are using recent AD-CD systems (or another z/OS package with OSA devices defined), we suggest you use QDIO mode because this represents the future direction for z/OS LAN operations.

3.7 Useful networking commands

The following commands might be useful when working with LAN devices:

**z/OS operator commands:**

- **D U,,ddd,nn**
  - dddd = address, nn = number to display
- **D M=DEV(dddd)**
  - provides path status
- **D M=CHP**
  - display all CHPIDs defined to z/OS
- **D IOS,MIH**
  - display current MIH values
- **SETIOS MIH,DEV=E201,TIME=00:30**
  - example of setting MIH

**TSO commands:**

- **NETSTAT DEV**
  - display all devices and links
NETSTAT HOME  display home address
NETSTAT GATE  display gateway addresses
NETSTAT CONN  display connection status
TRACERTE ipaddress
PING ipaddress

(Issue ALLOC DD(SYSCTPT) DA(*) before TRACERTE or PING for more data.)

VTAM commands:
V NET,ACT,ID=LCL701 vary local 3270 active to VTAM
D NET,MAJNODES display major nodes
D NET,ID=xxxxxx,E display information about specific node
D NET,TRL list the TRLEs
D NET,TRL,TRLE=OSATRL1E data about specific TRLE
V NET,ID=OSATRL,ACT activate a major node
V NET,ID=OSATRL,INACT
V NET,ID=ISTTRL,ACT,UPDATE=ALL remove inactive TRLEs from TRL list

Note that the name LCL701 in the sample V NET command is the VTAM name of the terminal. This name is not related to the LUname specified in the zPDT devmap. A 3270 session has both an aws3274 LUName (specified in the zPDT devmap) and a VTAM name (specified in VTAMLST). Basic MVS operator consoles are not specified in VTAM and have no VTAM name. This terminology is unfortunate because the aws3274 LUName (used to link a TN3270e session to an aws3274 definition) is not necessarily the same LUName associated with a VTAM operation.

Note that zPDT does not support the VMAC function from z/OS. The only virtual mac supported is generated on z/VM with the layer-2 vswitch.
Using basic zPDT commands

The full set of zPDT commands, with their syntax, is listed in Volume 1 of this series. This chapter illustrates use of the more common commands.

The zPDT commands fall into several categories:

- Setup commands, typically used when zPDT is not operational
- Basic operation commands to start zPDT, IPL a system, and later stop zPDT
- CP commands for low-level operations such as displaying System z memory
- Miscellaneous commands

All of these commands are Linux executable files. The operational commands must be used under the same Linux userid that started zPDT operation. Only one zPDT instance may be started by a given userid, and the commands assume they are to work with that instance. (The setup commands are not directly related to a running zPDT and can be used by any Linux userid that can access them.)
4.1 Setup commands

The following commands (with operand examples) are typically used when zPDT is not operational. The `awsckmap` command may be used after changing a devmap. It simply verifies that the devmap is in the correct format. It does not verify that files are in the correct emulated format.

```
$ awsckmap aprof1  # check devmap
$ awsckmap /bill/mapl23.txt  # can specify a full path name
$ awsckmap aprof1 --list  # long list
```

An emulated disk volume must be created before it can be used by zPDT. The `alcckd` command creates 3380 or 3390 volumes. There are four standard-size 3390 volumes (3390-1, -2, -3, -9) that can be created by specifying the full model type. Otherwise the `-s` parameter may be used to specify the number of cylinders to create.\(^1\) The emulated volume contains only the effective cylinders; diagnostic and alternate cylinders are not emulated.

```
$ alcckd /z/WORK01 -d3390-3  # create emulated 3390-3 volume
$ alcckd /z/WORK02 -s9000 -d3390  # create volume with 9000 cylinders
$ alcckd /z/WORK03 -d3390-1 -z  # create and zero the volume
```

The `alcckd` command does not place any software blocks (label, VTOC) on the volume. The user must do this later, using operating system utilities.

The `find_io` command (which has no operands) is used to list Ethernet interfaces and their associated CHPID number (relevant only for interfaces used for OSA Express emulation).

The `z1090ver` or `z1091ver` command may be used to determine the version of zPDT that is being used. The `z1090instcheck` command may be used to partly verify that the Linux environment is correct for zPDT.

4.2 Basic operation

These two commands start and stop zPDT operation. In the example, the devmap (file name `aprof1`) is in the current directory; otherwise, a complete path name would be used.

```
$ awsstart aprof1  # (start zPDT operation\(^2\))
$ awsstart aprof --clean  # (erase old zPDT logs/traces/dumps)
$ awsstart aprof13 --localtoken  # (ignore license server; use my token)
$ awsstop  # (stop zPDT operation)
```

The `awsstop` command instantly ends System z operation. No warning is sent to the System z operating system. We recommend that the System z operating system be stopped normally (whatever this might mean for various operating systems) before issuing `awsstop`.

The `token` command may be used to verify that the hardware key is available to zPDT. This command works only when zPDT is running.

The `ipl` command performs the architected System z IPL operation. It must point to a device that has a suitable IPL program installed. This is normally the operating system residence device, but can be a disk or tape that contains a stand-alone System z program. The use of an IPL parameter is completely dependent on the operating system involved and how the

---
\(^1\) Specifying more than 65520 cylinders creates an EAV volume. Verify that your System z operating system supports these before creating one.

\(^2\) The earlier versions of `awsstart` required the characters "--map" before the devmap name. This is now optional.
parameter is configured for that instance of the operating system. The zPDT environment must be initialized (using `awsstart`) prior to issuing the IPL command.

```
$ ipl 560  # (simple IPL)
$ ipl A80 parm 0A82CS  # (IPL with parameter)
```

The `awsmount` command is used while zPDT is running. It displays or changes the emulated device file associated with an emulated device. The first example is equivalent to mounting a tape on the device at address (device number) 560. This function makes the device ready and sends the appropriate attention interrupt to the operating system. This command is not limited to emulated tape drives, although that is the most common use. It can also be used for emulated disk drives or an emulated printer.

```
$ awsmount 560 -m /z/mytape  # (mount volume when none is mounted)
$ awsmount 560 -o /z/tape05  # (mount another volume, replacing old)
$ awsmount 560 -q  # (query device file status)
$ awsmount 560 -s  # (rewind emulated tape)
$ awsmount 560 -x  # (rewind and unload emulated tape)
```

If a file name is mounted as an emulated tape and that file does not already exist, it is automatically created by the `awsmount` command. Use care when typing the file name in an `awsmount` command to ensure that you specify the correct name and directory. For example, suppose you intended to mount and write to an existing emulated tape, `/tape/oldtape1`, but instead typed:

```
$ awsmount 560 -m oldtape1
```

This would allocate a new awstape file named `oldtape1` in your current directory. If you then write a large amount of data to this file, it can fill your `/home` file system and possibly cause problems. The z/OS label processing routines will catch some of these problems but not all, so ensure that you use the correct names.

Files mounted with `awsmount` must be appropriate for the device. A file mounted on an emulated tape drive must be in awstape format. (For an emulated tape drive opened for output, a new awstape file is automatically created if necessary.) A file mounted on an emulated 3390 disk drive must be in awsckd format. None of these formats are standard Linux formats; the emulated disk device files must be created with zPDT utilities. Emulated tape files (awstape) are created by writing to an emulated tape drive, mounted with the `awsmount` command.

The `awsstat` command provides the status of emulated I/O devices, including the Linux file name mounted on the device and (for disks) the last track used.

```
$ awsstat  # (query status of all emulated devices)
$ awsstat 0A80 0A82  # (query status of specified devices)
$ awsstat -i2 0a82  # (display status every 2 seconds)
```

The display output from `awsstat` is wide; it is best to widen the Linux window to almost full width before using `awsstat`. An example of the command is shown here:

```
$ awsstat A82
  Config file: /ibmsys1/aprof1
  DevAdr Subch ---Mgr-- Actv Busy --PID-- -----Device Information------
   0A82  15  AWSCKD  Yes  No  4329  Cyl-368, Head-10  /z/S7SYS1
```

Most of the output has obvious meanings. The `Subch` (subchannel) is an internal detail and the `PID` (Linux process id) is not normally needed. The `DevInfo` field shows the current logical position in the device.
The **awsin** command sends input to an emulated 3215. z/OS does not use 3215 devices, but they are sometimes used by other operating systems. If only one 3215 is defined, then the device number (009, in this example) does not need to be entered (unless the data element starts with a numeric digit; in this case the device number must be entered).

```
$ awsin 009 'This is input'  
(assume 3215 address 009)
$ awsin 'This is input'      
(if only one 3215 defined)
```

The **oprmsg** command sends input to the “hardware console” or “HMC console”, or “SCLP operator message interface.” This is the interface used by z/OS, for example, when all the normal z/OS consoles have been lost. There is no device address associated with this interface.

```
$ oprmsg 'V CN(*),ACTIVATE'  
(send message to z/OS)
```

zPDT commands function as Linux commands, and are entered through a shell prompt. The operands must conform to Linux shell rules. Some characters, such as the dollar sign ($), the right parenthesis ), and the left parenthesis ( have special meanings when entered as part of a command through a shell. It is necessary to escape these characters. This can be done by placing single quotes (’ ) around the whole operand or by placing a backslash (\) before each special character.

The **ready** command generates an asynchronous device end for the indicated emulated device. This is typically associated with mounting a new volume on the device.

```
$ ready 590
```

The **ipl_dvd** command emulates the operation of using an IPLable DVD in an HMC. Some software, such as z/VM, is distributed in this way.

```
$ ipl_dvd /media/530_GA_3390_DASD_DVD/cpdvd/530vm.ins
```

The file name used with **ipl_dvd** must be in the correct .ins format, which is similar to an OMA control format. Our particular example involves VM, and this might not work without the HMC function because this mode of VM installation uses unique HMC 3270 functions. Some Linux distributions might be packaged for DVD installation.

### Search paths

Our examples use a parameter on the **awsstart** command to specify the devmap. There are other ways to find these files. The complete search algorithm, in the order used, is shown here.

To find the devmap:

- Use the file named in the awsstart parameter, if specified.
- Use the CONFIG_FILE environment variable, if it exists.
- Use file devmap.txt in the current directory, if it exists.
- Use /home/<userid>/z1090/configs/devmap.txt, if it exists (not recommended).
- Error, if none of the preceding items exist.

### 4.3 CP commands

The following commands control operation of zPDT System z processors, known as CPs. The first (or only) processor is CP number 0. A second processor (if it exists) is CP number 1, and so forth. By default, these commands are sent to CP 0. The **cpu** command may be used to
specify a different default processor. These commands can also be directed to zIIPs, zAAPs, and IFLs which are assigned cpu numbers as though they were CPs.

$ cpu 1 
    (specify default processor)
$ stop 
    (stop indicated (or default) processor)
$ start 1 
    (start indicated (or default) processor)
$ restart 0 
    (perform CP restart function)
$ query 0 
    (query indicated processor)
$ interrupt 0 
    (send external interrupt)
$ storestatus
$ sys_reset 
    (perform System z reset)
$ storestop 1AC0 
    (stop on store in absolute address)

$ adstop 1AC0 on  
    (set address stop at indicated address)
$ adstop 1AC0 off 
    (remove the address stop)
$ memid <file name> address 
    (load Linux file into System z memory)
$ loadparm 0a8200 
    (set SCLP IPL parm (for SCLP usage only)
$ loadparm -d 
    (display SCLP IPL parameter)

The following display and change commands are best used when the CP is stopped. Addresses specified for the adstop, d, and st commands are real addresses or virtual addresses, depending on how the address is written:

address 1AC0 
    (real memory address)
address v1AC0 
    (virtual memory address (primary AS))
address vh1AC0 
    (virtual memory address (home AS))
address vs1AC0 
    (virtual memory address (secondary AS))

$ d v1AC0 64 
    (display 64 bytes from address 1AC0)
$ d 1AC0.40 
    (display x'40’ bytes (real address))
$ d t vh1AC0 32 
    (display, including EBCDIC)
$ st v1AC0 47F01CD0 
    (modify memory value at address v1AC0)

$ d psw 
    (display psw for current mode)
$ st psw FF007AB0 0 0 123456 
    (set 128-bit PSW)

$ d g 
    (display general registers)
$ d g3 
    (display general register 3)
$ st g1 1234 
    (store x'00001234’ in 32-bit register)
$ st gx1 1234 
    (store x'0000000000001234’ in 64-bit reg)

$ d y 
    (display floating point registers)
$ d y3 
    (display floating point register 3)
$ st y1 1234567887654321 
    (store x’1234567887654321’ 64-bits)

$ d z 
    (display access registers (32-bits))
$ st z3 ABCD 
    (store x'0000ABCD' in access register 3)

$ d x 
    (display control regisers)
$ st xl AABBCCDD 
    (store 32 bits in control register 1)
$ st xx1 AABBCCDD 
    (store 64 bits (x'00000000AABBCCDD’)

$ d r 
    (display architecture mode)
$ d yc 
    (display floating point control register)

3 By “real memory address” we mean an address in System z memory. The “real address” is not translated through the System z virtual memory process.
The following conventions are used for the display and store commands:

- The display storage commands take the specified address and round it downward to the nearest 32-byte (hex 20) boundary.
- Memory display commands display 32 bytes (in hex) per line. If the t parameter is used, EBCDIC is also displayed at the end of each line. In either case, the lines are long and the Linux window used should be wide.
- Store commands for memory work in bytes, starting at the address specified in the st command.
- Registers are displayed as 64-bit registers when in z/Architecture mode and 32-bit values when not in z/Architecture mode.
- Store commands (memory and registers) assume data is in hex.
- Store commands for registers work in 32-bit words (st g commands) or 64-bit words (st gx commands). Data is padded on the left with zeros.
- Memory displays include the storage protection key (displayed after the hex data).
- PSW values (st psw xxx) are entered as 32-bit words. Changing a 128-bit PSW means that four operands are needed. The command st psw 123 456 789 ABC places x’00000123’ in the first word of the PSW, x’00000456’ is intended for the second word (but the second word is forced to zeros), x’00000789’ is placed in the third word, and x’00000ABC’ is placed in the last word.
- If a storage word operand overflows (for a register or segment of the PSW), the value is set to x’FFFFFFFF’.
- Floating point registers are always treated as 64-bit words.

A memld command might be:

```
$ memld /home/ibmsys1/initrd 800000
```

This would read the contents of the indicated Linux file into the System z memory, starting at real address (hexadecimal) 800000.

### 4.4 Devmaps

A devmap must exist when starting zPDT operation. The devmap defines the size of System z memory and several other parameters. It also specifies which device managers to start and what emulated devices are connected to each device manager.

Many devmaps may exist; they are simple Linux text files. Only one devmap can be used when starting a zPDT instance, and it cannot be dynamically changed while zPDT is running.

Typically, zPDT users have a number of devmaps for a variety of purposes. For example:

- A devmap for the current z/OS release, with only the basic IPL volumes and no OSA
- A devmap for the current z/OS release, with all the volumes that have been installed and with one or two OSAs (depending on the LAN and tunnel environment desired)
- A devmap for a previous z/OS release (using the normal AD-CD device numbers)
- A devmap contains both the current AD-CD release (using the normal device numbers) and a previous release (using other device numbers)
- A devmap for basic z/VM
A devmap for z/VM and one (or two) z/OS releases

Appendix A, “z/OS 1.12 AD-CD example” on page 61, provides an example of a complete devmap, along with the z/OS TCP/IP setup required for both LAN and tunnel usage.
Frequently asked questions

The following FAQs are more specific to installation and detailed usage. Volume 1 of this book series contains additional FAQs that are more related to initial zPDT concepts.

Q: Can I use any Linux userid instead of ibmsys1?
A: You can use any Linux userid for zPDT that is not longer than eight characters. You should use the same userid for zPDT installation and operation.

Q: Can a solid-state disk drive be used (in the PC) instead of a traditional hard disk?
A: Yes.

Q: You say that Ethernet SNA operation is not supported. Might it work?
A: Yes, it might. It has not been tested and IBM will not respond to problems using it.

Q: When downloading 3390 volumes the file name is sometimes changed, adding a bin suffix. Why?
A: This seems to be a characteristic of the Chrome browser. Try a different browser.

Q: Can I use a SCSI DLT tape drive?
A: It should work (if it supports the SSC-3 SCSI Command Set for Sequential Devices), although this is not supported by IBM and has not been tested.

Q: Can I use a SCSI 4 mm tape drive?
A: It might work but we strongly suggest you do not use 4 mm drives. These have proven to be poorly suited for emulated IBM S/390® work.

Q: Will using a zIIP or zAAP or IFL increase the performance of my zPDT?
A: No, assuming you are replacing a CP with the zIIP or zAAP. These specialty processors operate at the same speed as a “normal” zPDT CP. They are provided to allow developers to verify that their applications use a zIIP or zAAP or IFL in the intended manner. It is possible they could improve performance through parallel operation with the base CP.
Q: Is the floating point hardware of the base machine used when executing System z floating point operations?
A: Sometimes. This is a complex area when floating point exception conditions are considered and IBM has not disclosed zPDT design for this area.

Q: Why do you not provide a definite MIPS value? This would help us determine how to best use zPDT.
A: There are no definite numbers. A MIPS measurement is very dependent on the exact workload. The approximate ranges we sometimes discuss for zPDT assume a traditional commercial application mix, although even this is difficult to define exactly.

Q: Can I have multiple tap (tunnel) interfaces, such as tap0, tap1, and so forth?
A: Yes, starting with the zPDT release in Spring 2010. A total of four OSA devices of any type may be defined. The tunnel interfaces will typically have CHIPD numbers A0, A1, A2, and A3. The tap devices may be defined (as seen by the `find io` command), but they are not used unless a corresponding OSA device exists in the devmap.

Q: Can I run multiple TCP/IP stacks on a single emulated OSA-Express adapter?
A: Yes.

Q: Can I place emulated 3390 volumes on an NFS server?
A: In principle, yes. However, access time must remain within the tolerance that z/OS expects for disk activities. In general, we do not think this is a good idea. (If any I/O operation takes more than 30 seconds, the zPDT device manager is considered dead and is restarted. z/OS has internal timers with much shorter timeouts than 30 seconds.)

Q: Can I share emulated volumes on an NFS server with several zPDT machines?
A: No. This would almost certainly result in corrupted data unless the NFS files are read-only files. Such usage might be practical for `awstape` files. Read-only DASD is not supported by z/OS.

Q: Why might I need to specify a unit address in the device statements for OSA? I do not understand these.
A: A full discussion is beyond the scope of this document. For TCP/IP, you need to ensure that the unit addresses associated with non-QDIO TCP/IP usage are 0 and 1. (This is required by the default OAT used by OSA.) You need to ensure that unit address FE is used only for OSA/SF when using the default OAT. You need to remember that the default unit address is the same as the low-order two digits of the device number ("address"). If you meet these requirements, there is no need to specify a unit address in the device statements for OSA. For QDIO operations, you should not need the unit address operands.

Q: Does zPDT support thin interrupts?
A: Yes, for OSA devices and cryptographic emulation. (This function is properly known as the Adapter Interrupt Facility.)

Q: Can I filter IP traffic before it is sent to my emulated OSA-Express interface? This reduces the overhead involved in rejecting packets not addressed to my system.
A: In OSD (QDIO) mode, there is some automatic filtering. In OSE (non-QDIO) mode, you can customize the OAT with your IP address. If this is done, the OSA interface will pass only packets intended for this IP address. If this customization is not done (and it is not done in the default OAT), then all packets are sent to the host TCP/IP and unwanted packets are rejected at that level. If you use NAT functions on the base Linux, then most of the filtering is done at that level.

Q: Is OSA-Express emulation different than OSA emulation?
A: Yes, very much so, although OSA operation of simple TCP/IP can usually be provided by OSA-Express without changes to the System z operation. (This question is related to a
terminology problem and assumes that OSA means the original OSA adapter, which operated largely as an LCS device.)

Q: Is OSN operation (CDLC) provided with OSA-Express2 emulation?
A: No.

Q: Does OSA-Express2 emulation support jumbo frames? With QDIO? With non-QDIO?
A: Jumbo frames are not supported by zPDT with QDIO, although some users have been successful using them. Appropriate kernel parameters are needed as specified in “Notes for sysctl values” on page 9.

Q: Should I use 1492 or 1500 as the maximum packet size (MTU) when using awsosa?
A: We use a maximum of 1492. The details are beyond the scope of this document. (As best we can tell, the System z communication routines automatically adjust this number down if necessary. Thus it probably does not matter whether you specify 1492 or 1500.)

Q: I want to use the OSA Express function with QDIO, but my z/OS does not have any OSA devices defined. How do I add them to z/OS?
A: This is done with the HCD utility under z/OS, creating a new IODF. Extensive IBM documentation exists for HCD. An example is included in the third book in this series (SG24-7723).

Q: Does QSA-Express emulation include advanced functions such as VIPA?
A: Yes, when using QDIO.

Q: Can I use a continuing range of addresses (device numbers) when I have multiple OSA QDIO interfaces? For example, 400-402 for TCPIP1, 403-405 for TCPIP2, and so forth.
A: No. For z/OS we believe the first OSA address (for a TCP/IP stack, in this example) must be an even number. You would need to use 400-402, skip 403, then use 404-406, skip 407, and so forth. (This statement might not be correct for z/VM.)

Q: Do the OSA-Express2 offload functions work? Do they accomplish anything on an emulated system?
A: The Linux-based zPDT OSA implementation does not use offload functions at this time. In some cases (with the latest Linux kernels) you might need to force Linux to disable offloading.

Q: What PC Card (PCMCIA-type card) should I use for additional Ethernet ports on a ThinkPad?
A: Use any card that the base Linux system accepts. We tested with an Xterasys Gigabit PC Card (98-012084-585). We also informally tried several older IBM 10/100 Etherjet cards.

Q: Can I use IP aliasing in Linux while using zPDT?
A: Yes, but the alias addresses are not relevant to zPDT and are not displayed by the find_io command.

Q: The find_io command indicates a MAC address of 0:0:0:0:0:0 for the tap0 interface. Can a MAC address be zeros?
A: This is normal for the tap interfaces. A “real” LAN adapter should not have a zero MAC address. Later versions of zPDT create artificial MAC addresses for tap interfaces.

Q: I have multiple Ethernet adapters, each on a different subnet. Response is very slow and I get multiple responses to pings. Is there a problem using multiple adapters?
A: In general, no. However, multiple interfaces on different subnets should not be connected to the same VLAN. This creates routing, ARP, and duplicate response issues. Also, the external routing configuration (external to your system) might produce multiple responses. Multiple subnets on a single physical network might produce multiple responses.
Q: I have an error message about GVRP when I try to use a VLAN/VSWITCH in z/VM. Is this supported?  
A: No, GVRP is not supported. Specify NOGVRP for your VSWITCH. VLAN generally works, but there are exceptions. See the note in Chapter 10 of the third book in this series (SG24-7723-04).

Q: Why do some AD-CD releases pause for many seconds while shutting down?  
A: You can edit the SHUTDOWN entries in PARMLIB to remove or change any pause statements.

Q: I am using z/VM and have a problem with MONITOR. Does zPDT support this usage?  
A: An APAR fix for z/VM 5.2 and 5.3 is available to resolve this problem. It is APAR VM64385.

Q: You are inconsistent with the addresses for the AD-CD volumes. For example, sometimes volume SARES1 is at address A91 and sometimes at address AA0. Which is correct?  
A: Both are correct. Any 3390 volume can be at any address that is defined as a 3390 in the IODF for that z/OS system. For ease of documentation we always show the IPL volume at A80 and the SYS1 volume (which contains the IODF and IPLPARM datasets) at A82, but these addresses are not required. The IPL address and parameter much match the addresses you use.

Q: What happens if I remove the hardware key?  
A: The zPDT will stop after a while.

Q: You use userid **ibmsys1** throughout all the examples. Is there something special about this userid?  
A: No.

Q: Is there any national language support in zPDT?  
A: No.

Q: Can I use an alternate translation table to convert EBCDIC to ASCII for awspart output?  
A: No.

Q: Can I use multiple zPDT tokens to obtain more CPs?  
A: Yes, but your number of CPs might be limited by other aspects of your zPDT license.

Q: I have several Linux windows open while running zPDT. I can enter zPDT commands in any window, which is convenient. However, I also sometimes get output messages in a different window from where I entered a command. Is this normal?  
A: Yes. zPDT output messages (but not command output messages) are sent to the console session that issued the **awstart** command.

Q: My z1090 rpm installation failed with an error message about db_recovery. What now?  
A: Try the command **rpm --rebuilddb** and then install z1090 again (using the z1090 installer program, and not trying to directly install the z1090 rpm).

Q: Can I routinely migrate to the next Linux releases when they become available?  
A: Maybe. There is no unique zPDT tie to a particular release. However, it is possible that the zPDT installation steps might not work for a new release (due to different library paths) or that the new release might not support the particular hardware in your base machine. Consult your zPDT provider.

Q: I am using an emulated printer and this sends output to a Linux file. Does this file remain open for output by zPDT all the while zPDT is running?  
A: Yes. It is closed if you use **awsmount** to assign a new output file for the printer.
Q: Is there an easy way to delete all the existing disk partitions when installing openSUSE?
A: Yes, delete the `/dev/sda` entry. This does not really delete the hard disk, but it does delete all partitions on it.

Q: Can emulated printer output be directed to `/dev/lp0` or something similar?
A: We do not know; this was not tested.

Q: Does zPDT operate in kernel mode? In suid mode?
A: Kernel mode is not used, but one module (part of awsoa) operates in suid mode.

Q: How do I add more SUSE components after I have completed the basic installation?
A: Application → System → Yast → Software Management → Filter: Package Groups and then follow the installation prompts. The exact navigation changes with new distributions, but the general idea remains the same.

Q: You specified that userid `ibmsys1` should be a member of group zpdt. Is this needed?
A: No. However, it is useful for additional Linux security measures as described in the third book in this series (SG24-7723).

Q: Are zPDT commands case sensitive? Can I issue `ipl` or `IPL`?
A: The commands are case sensitive. They are simply the names of Linux files and Linux file names are case sensitive.

Q: Can I run as `root` when installing and using zPDT?
A: No. Follow the instructions concerning when to work as root and when to work under a normal userid (such as `ibmsys1`).

Q: The `z1090instcheck` command does not work or gives the wrong results. Why?
A: You might need the full path name for the `z1090instcheck` command in some cases. Your `PATH` might be pointing to a down-level version at the time you issue the command. Also, you must have installed zPDT code before you can use this command.

Q: I have a SCSI tape drive. I want to use it directly for Linux functions (not directly connected with zPDT operation) but I cannot find the `mt` command (a "standard" Linux command for manipulating tape devices).
A: We noticed that `mt` is not always installed with some Linux distributions. In some cases it appears to be part of the `cpio` rpm.

Q: Does zPDT handle 3270 nulls correctly?
A: This is not a function of zPDT; it is a function of the 3270 emulator and, to some extent, the application involved. Relevant functions for x3270 can be found in Options → Toggles → Blank Fill. The ISPF command `nulls on|std|all|off` might be relevant.

Q: Does IBM need to enable something to allow full operation of the System z internal CRYPTO instructions?
A: No, full operation is always enabled.

Q: How can I write a tape mark on an AWSTAPE volume?
A: Use `awsmount xxx --wtm` where `xxx` is the address (device number) of the tape drive. Notice there is a double dash before the `wtm` option.

Q: Where do I obtain OMA distributions?
A: We do not know of any IBM products currently distributed in OMA format. Some Linux for System z distributions might be in this format.

Q: Should device statements (in a devmap) be in order by addresses?
A: No particular order is needed.
**Q:** I want to use PCOMM instead of x3270. Is this acceptable? Can you include it with the zPDT package?

**A:** Yes. However, you should use a release later than PCOMM 5.5. We have verified that version 5.5 is not suitable for zPDT. PCOMM is part of a separate IBM product. We cannot include it as part of the zPDT package.

**Q:** I installed the recommend Linux on my T61p ThinkPad, but I cannot access the token (or a USB disk drive).

**A:** It appears that USB usage might be sensitive to BIOS levels. In at least one case, updating the T61p to the current BIOS level (which was 2.09, at the time this was noticed) solved the problem.

**Q:** The AD-CD system always starts TCP/IP and associated jobs. How can I delete them?

**A:** You can edit the VTAMAPPL types of entries in PARMLIB and remove the associated commands. While running z/OS you can issue P TCPIP, wait a few seconds, and then issue C INETD4.

**Q:** Can I use RMF?

**A:** Yes, but not all of it is relevant on a zPDT system.

**Q:** I have volumes at addresses A80 through A8F. Do I need to define a new awsckd unit in order to add more disk volumes?

**A:** No, you can have up to 256 volumes in one instance of awsckd.

**Q:** I am using the IBM Personal Communications product to connect from a remote PC to z/OS running on zPDT. Every time I start Personal Communications it wants to print something. How can I stop this?

**A:** This is a well-known issue, and is not related to zPDT. Personal Communications stores user profiles in .ws files (such as bill.ws, for example). Find the .ws profile you are using and add the following lines at a reasonable place in the profile:

```
[LT]
IgnoreWCCStartPrint=Y
```

**Q:** Can I make the kernel.shmmax value very large to avoid worrying about it?

**A:** As far as zPDT is concerned, you could do this. However, it is possible that other Linux applications might accept the very large value and attempt to use unreasonable amounts of shared virtual memory, resulting in excessive paging.
z/OS 1.12 AD-CD example

Attention: The current z/OS AD-CD at the time of writing is for release 1.13. We use z/OS 1.12 in the following material because it has fewer volumes. This tends to make the sample setup clearer. The purpose here is to illustrate the overall setup for a zPDT system, and not to explore options of a particular AD-CD z/OS release.

We installed AD-CD z/OS 1.12 AD, including all the volumes present on the DVDs. We used the single ThinkPad Ethernet adapter for OSA connectivity and configured it for QDIO mode. This appendix describes our specific definitions. Although this material applies only to the z/OS 1.12 AD-CD system, it might be helpful when setting up other environments.

The examples we use assume a machine with modest memory (4 to 8 GB). A machine with larger memory would probably have larger root and swap partitions.
A.1 Disk planning

All the 3390-3 volumes in the 1.12 release require about 87 GB of disk space. (It is unlikely that many users would elect to install all the volumes. Four volumes are needed to IPL; all the others are optional, depending on your usage plans.) We also had two 3390-1 volumes with local data. This made a total of about 90 GB for emulated disk volumes. (Note that later z/OS AD-CD releases require considerably more disk space if all volumes are installed. For the z/OS AD-CD 1.13 release, installing all volumes requires about 140 GB.)

Our Linux base was installed in a single 12 GB partition, and we created a 4 GB Linux swap partition.

The AD-CD z/OS 13 distribution is considerably larger, especially for DB2 volumes. The purpose of the following listings is to show complete zPDT and z/OS definitions, and the z/OS 1.12 is more compact for this purpose.

A.2 Connectivity planning

We had a single Ethernet adapter in the Lenovo W520 we used. For our LAN setup we used Scenario 4 described earlier. This scenario includes a tunnel OSA connection between the base Linux and z/OS, plus an OSA LAN connection from z/OS to the outside world. We also used aws3274 connections (which appear as local, channel-attached devices) for the MVS console and simple VTAM terminals. The TCP/IP connectivity is as follows:

- The IP address for the base Linux (from external connections on a router/hub) is 192.168.0.80. This IP address cannot be accessed from z/OS.
- The IP address of z/OS (from external connections on a router or hub) is 192.168.0.81. This IP address cannot be accessed from Linux. (This is not a valid IP address for connection to the Internet; you must provide your own fixed IP address.)
- The IP address for internal connections to Linux TCP/IP (from the x3270 sessions, for example) is 127.0.0.1; this is the default localhost address for Linux.
- The IP address used to access z/OS from Linux (via tunnel) is 10.1.1.1.
- The IP address used to access Linux from z/OS (via tunnel) is 10.1.1.2.
- The IP address of the router is 192.168.0.1; this is the default route for z/OS (and Linux).

A.3 Listings

We used the find_io command to determine that our Ethernet adapter was eth0, and that it was assigned as CHPID F0. (As a practical matter, the integrated Ethernet adapter is usually assigned to CHPID F0.) The tunnel interface is usually CHPID A0. We elected to use the QDIO mode for both OSA interfaces. We used the following devmap:

```
[system]
memory 6000m
3270port 3270
processors 2

[manager]
nname aws3274 0002
device 0700 3279 3274
device 0701 3279 3274
```
device 0702 3279 3274
device 0703 3279 3274

[manager]
name awsckd 0001
device 0A80 3390 3990 /z/ZCRES1
device 0A81 3390 3990 /z/ZCRES2
device 0A82 3390 3990 /z/ZCSYS1
device 0A83 3390 3990 /z/ZCUSS1
device 0A84 3390 3990 /z/ZCPRD1
device 0A85 3390 3990 /z/ZCPRD2
device 0A86 3390 3990 /z/ZCPRD3
device 0A87 3390 3990 /z/ZCDIS1
device 0A88 3390 3990 /z/ZCDIS2
device 0A89 3390 3990 /z/ZCDIS3
device 0A8A 3390 3990 /z/ZCDIS4
device 0A8B 3390 3990 /z/ZCDIS5
device 0A8C 3390 3990 /z/ZCDIS6
device 0A8D 3390 3990 /z/ZCD91
device 0A8E 3390 3990 /z/ZCD92
device 0A8F 3390 3990 /z/ZCD93
device 0A90 3390 3990 /z/ZCD981
device 0A91 3390 3990 /z/ZCD982
device 0A92 3390 3990 /z/ZCD983
device 0A93 3390 3990 /z/ZCD984
device 0A94 3390 3990 /z/ZCCI31
device 0A95 3390 3990 /z/ZCCI41
device 0A96 3390 3990 /z/ZCWEA1
device 0A97 3390 3990 /z/ZCWEA2
device 0A98 3390 3990 /z/ZCWEA3
device 0A99 3390 3990 /z/ZCWEA4
device 0A9A 3390 3990 /z/ZCWEA5
device 0A9B 3390 3990 /z/ZCWEA6
device 0A9C 3390 3990 /z/ZCWei1
device 0A9D 3390 3990 /z/ZCWei2
device 0A9E 3390 3990 /z/SARES1
device 0AA0 3390 3990 /z/WORK01 #local volumes, not part of AD-CD
device 0AA1 3390 3990 /z/WORK02

[manager]
name awsoa 0019 --path=A0 --pathtype=OSD --tunnel_intf=y
device 400 osa osa
device 401 osa osa
device 402 osa osa

[manager]
name awsoa 0009 --path=F0 --pathtype=OSD
device 404 osa osa
device 405 osa osa
device 406 osa osa

[manager]
name awstape 004
device 581 3490 3490
No modifications or definitions are needed in Linux to define the tunnel interface. This interface (and associated IP address) does not exist until `awsstart` is issued, using a devmap that contains the tunnel_intf operand in an awsosa definition.

QDIO operation requires a TRL major node in VTAM. This is provided by member OSATRL1 in ADCD.Z112.VTAMLST in the AD-CD system, which we changed slightly. (The TRL member name is included in the list in member ATCCON00). The continuation characters (the X characters) are in column 72.

```
OSATRL1 VBUILD TYPE=TRL
OSATRL1E TRLE LNCTL=MPC,READ=(0400),WRITE=(0401),DATAPATH=(0402), X
   PORTNAME=PORTA, X
   MPCLEVEL=QDIO  
OSATRL2E TRLE LNCTL=MPC,READ=(0404),WRITE=(0405),DATAPATH=(0406), X
   PORTNAME=PORTB, X
   MPCLEVEL=QDIO
```

We need both TRLEs only when using two OSA definitions.

We used the following TCP/IP profile in z/OS. (If using the z/OS AD-CD system, be certain to update the correct PROFILE. In the z/OS 1.12 AD-CD system, this is member PROF1.)

```
ARPAGE 5
DATASETPREFIX TCPIP

AUTOLOG 5
   FTPD JOBNAME FTPD1 ; FTP Server
   PORTMAP ; Portmap Server
ENDAUTOLOG

PORT
   7 UDP MISCSERV ; Miscellaneous Server
   7 TCP MISCSERV
   9 UDP MISCSERV
   9 TCP MISCSERV
   19 UDP MISCSERV
   19 TCP MISCSERV
   20 TCP OMVS NOAUTOLOG ; FTP Server
   21 TCP OMVS ; FTP Server
   23 TCP TN3270 ; Telnet Server
   25 TCP SMTP ; SMTP Server
   53 TCP NAMESRV ; Domain Name Server
   53 UDP NAMESRV ; Domain Name Server
   69 UDP OMVS ; OE TFTP SERVER
   80 TCP OMVS ; OE WEB SERVER
   111 TCP PORTMAP ; Portmap Server
   111 UDP PORTMAP ; Portmap Server
   135 UDP LLBD ; NCS Location Broker
   161 UDP OSNMPD ; SNMP Agent
   162 UDP SNMPQE ; SNMP Query Engine
   433 TCP OMVS ; OE WEB Server
   443 TCP OMVS ; Secure Server
   512 TCP RXSERVE ; Remote Execution Server
```
513 UDP OMVS ; OE RLOGIN SERVER
514 UDP OMVS ; OE syslog server
514 TCP RXSERVE ; Remote Execution Server
515 TCP LPSERVE ; LPD Server
520 UDP OROUTED ; RouteD Server
580 UDP NCPROUT ; NCPROUTE Server
750 TCP MVSKERB ; Kerberos
750 UDP MVSKERB ; Kerberos
751 TCP ADM@SRV ; Kerberos Admin Server
751 UDP ADM@SRV ; Kerberos Admin Server

; 1021 TCP OMVS ; OE FTP SERVER
1023 TCP OMVS ; OE TELNET SERVER
1023 UDP OMVS ; OE TELNET SERVER
1024 TCP OMVS ; OE SERVICES
1415 TCP CSQ1CHIN ; CSQ1 MQ TCP Listener
3000 TCP CICSTCP ; CICS Socket
32200 TCP BBN7ACRS ; Daemon IP port
32201 TCP BBN7ACRS NODELAYACKS ; Daemon SSL port
32202 TCP BBNS001 ; SOAP JMX Connector port
32203 TCP BBNS001 ; ORB port
32204 TCP BBNS001 NODELAYACKS ; ORB SSL port
32205 TCP BBNS001 ; Administrative console port
32206 TCP BBNS001 NODELAYACKS ; Administrative console secure port
32207 TCP BBNS001 ; HTTP transport port
32208 TCP BBNS001 NODELAYACKS ; HTTPS transport port
32209 TCP BBNS001 ; Administrative Local Port
32210 TCP BBNS001 ; High availability manager comm port
32211 TCP BBNS001 ; Service Integration port
32212 TCP BBNS001 NODELAYACKS ; Service Integration secure port
32213 TCP BBNS001S ; Service Integration MQ interop
32214 TCP BBNS001S NODELAYACKS ; Service Integration MQ inter secure
32215 TCP BBNS001 ; Session Initiation Protocol (SIP)
32216 TCP BBNS001 NODELAYACKS ; Session Init Protocol (SIP) secure

8879 TCP BBODMGR ; SOAP JMX Connector port
7277 TCP BBODMGR ; Cell Discovery port
9080 TCP BBODMGR ; ORB port
9090 TCP BBODMGR ; HTTP port
9043 TCP BBODMGR ; HTTPS port
5755 TCP BBODMNC ; Daemon port
5756 TCP BBODMNC ; Daemon SSL port

; WAS Base Node entries
8880 TCP BBOS001 ; SOAP JMX Connector port
9080 TCP BBOS001 ; ORB port (COMMENT THIS ONE OUT)
9443 TCP BBOS001 ; HTTP port
5655 TCP BBODMNB ; Daemon port
5656 TCP BBODMNB ; Daemon SSL port

; WAS ND Node entries
8879 TCP BBODMGR ; SOAP JMX Connector port
7277 TCP BBODMGR ; Cell Discovery port
9080 TCP BBODMGR ; ORB port
9090 TCP BBODMGR ; HTTP port
9043 TCP BBODMGR ; HTTPS port
5755 TCP BBODMNC ; Daemon port
5756 TCP BBODMNC ; Daemon SSL port
; WAS Federation entries
8878 TCP BBON001 ; SOAP JMX Connector port
7272 TCP BBON001 ; Node Discovery port
2809 TCP BBON001 ; Node Agent’s ORB port
9810 TCP BBOS001 ; Base Server’s ORB port

SACONFIG DISABLED

DEVICE PORTA MPCIPA
LINK ETH1 IPAQENET PORTA
HOME 10.1.1.2 ETH1

DEVICE PORTB MPCIPA
LINK ETH2 IPAQENET PORTB
HOME 192.168.0.61 ETH2

BEGINRoutes
; Destination SubnetMask FirstHop LinkName Size
ROUTE 192.168.0.0 255.255.255.0 = ETH2 MTU 1492
ROUTE 10.0.0.0 255.0.0.0 = ETH1 MTU 1492
ROUTE DEFAULT 192.168.0.1 ETH2 MTU 1492
ENDRoutes

ITRACE OFF

IPCONFIG NODATAGRAMFWD

UDPCONFIG RESTRICTLOWPORTS

TCPCONFIG RESTRICTLOWPORTS

START PORTA
START PORTB
Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 67. Note that some of the documents referenced here may be available in softcopy only.

- *IBM System z Personal Development Tool Volume 1 Introduction and Reference*, SG24-7721
- *IBM System z Personal Development Tool Volume 3 Additional Topics*, SG24-7723
- *IBM System z Personal Development Tool Volume 4 Coupling and Parallel Sysplex*, SG24-7859
- *Communications Server for z/OS V1R9 TCP/IP Implementation Volume 1: Base Functions, Connectivity, and Routing*, SG24-7532

Other publications

These publications are also relevant as further information sources:

- *z/Architecture Principles of Operation*, SA22-7832

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

*ibm.com/redbooks*

Help from IBM

IBM Support and downloads

*ibm.com/support*

IBM Global Services

*ibm.com/services*
Index

Symbols
.bashrc, changes 10
/etc/sysctl.conf 9
/home file system 3
/usr/z1090/bin 7

Numerics
11S and MTSN fields 12
3270 interfaces 26
3270 nulls 59
3270 sessions, starting 20
3270port parameter 26
32-bit support 4
3390 volumes, additional 22
4-mm tape drive 55

A
activation, token 12
AD system, device addresses 18
Adapter Interrupt Facility 56
AD-CD system, installation 17
adstop command 51
alcckd command 48
aliasing, IP 57
APAR VM64385 58
Application Development System 17
ath0 device 43
aws3274 device manager 26
awscld, number units 60
awsckmap command 48
awsin command 50
awsmount command 49
awsOSA performance 42
AWSPRT, translation table 58
awsstart 20
awsstart command 48, 58
awsstat command 49
awsstop command 21, 48
awstop, format 16

B
BIOS level 60
bonding, LAN 26

C
CD/DVD drive access 7
CHPID number 28
Chrome browser 55
command
adstop 51
alcckd 48
awsckmap 48
awsin 50
awsmount 49
awsstart 48
awsstat 49
awsstop 21
cpu 51
d 51
find_io 62
interrupt 51
ipcs 7
ipl 21
ipl_dvd 50
memld 51
mt 59
oprmeg 50
query 51
ready 50
restart 51
st 51
start 51
stop 51
storestatus 51
storestop 51
sys_reset 51
sysctl 7
ulimit 7
z1090instcheck 59
commands, package 47
core image files 3
cpu command 51
CRYPTO instructions 59

D
d command 51
db_recovery, rpm error 58
device addresses, for AD 18
DEVICE name 28
devmap 52
devmap, for AD 19
disk partitions, deleting 59
disk planning 3
disk usage layout 3
dmidecode rpm 4
downloading 3390 volumes 55
dual boot system 4

e
Ethernet adapters, multiple 57
Ethernet ports, PCMCIA 57
ethtool usage 42
external interrupt 51

© Copyright IBM Corp. 2009, 2013. All rights reserved.
QDIO operation, advantages 44
QDIO setup 27
query command 51

R
ready command 50
Redbooks Web site 67
Contact us viii
releases, new 13
resolver, z/OS 39
Resource Link 13
restart command 51
RHEL 32-bit installation 4
RMF 60
root partition 3
root userid 59
router personal 41

S
Scenario 2 31
Scenario 3 33
Scenario 4 35
Scenario 5 37
SCSI DLT tape drive 55
SCSI tape 59
search paths 50
Secure Update Utility 12
SEL protection 41
serial numbers, 1090 11
shk-server 8
shmmax value 9
shutdown, z/OS and 1090 21
SNA 27
SNA 3270 26
SNA operation 26, 44
sntl-sud 8
software media 16
solid-state disk drive 55
st command 51
start command 51
START name 28
stop command 51
storestatus command 51
storestop command 51
SUSE components, adding 59
swap partition 3
sys_reset command 51
sysctl command 7
sysctl parameter 42
SYSTCPT DD statement 45
System z software 15

T
tap (tunnel) interfaces 56
TCP/IP stacks, multiple 56
TCP/IP, starting z/OS 60
telnet session 43
telnet-server, selection 4
thin interrupts 56
TN3270E clients 5
TN3270e clients 5
TN3270e connections 26
token 48
token serial number 11
TRACERTE command 45
TRL entry 28
TRL major node 64
TSO commands, NETSTAT 45
tunnel connection 28
tunnel connection, usage 43

U
ulimit command 7, 10
unit address, OSA 56
Universal Time 4
Unsupported Function, 3270 21
upddecode tool 4

V
VIPA functions 57
VLAN, usage 57
VLAN/VSWITCH in z/VM 58
VMAC support 45
vsftpd, selection 4
vswitch support 45
VTAM commands 45
VTAMLST ATCCON00 28

W
wireless connection 28
wireless usage 43

X
x3270 fonts 6
x3270 installation 5
x3270, obtaining 5
x3270, startup 20
x3270, to z/OS TCP/IP 43

Z
z/OS 16
z/OS TCP/IP 26–27
z/VM 16
z/VSE 16
z1090 rpm 8
z1090instcheck command 11, 48, 59
z1090ver command 48
z1091ver command 48
zIIP or zAAP, performance 55
This IBM Redbooks publication introduces the IBM System z Personal Development Tool (zPDT), which runs on an underlying Linux system based on an Intel processor. zPDT provides a System z system on a PC capable of running current System z operating systems, including emulation of selected System z I/O devices and control units. It is intended as a development, demonstration, and learning platform and is not designed as a production system.

This book, providing specific installation instructions, is the second of three volumes. The first volume describes the general concepts of zPDT and a syntax reference for zPDT commands and device managers. The third volume discusses more advanced topics that may not interest all zPDT users. The IBM order numbers for the three volumes are SG24-7721, SG24-7722, and SG24-7723.

The systems discussed in these volumes are complex, with elements of Linux (for the underlying PC machine), IBM z/Architecture (for the core zPDT elements), System z I/O functions (for emulated I/O devices), and IBM z/OS (providing the System z application interface), and possibly with other System z operating systems. We assume the reader is familiar with the general concepts and terminology of System z hardware and software elements and with basic PC Linux characteristics.