Redbooks

ibm.com/redbooks

The Virtualization Cookbook for IBM Z Volume 2

Red Hat Enterprise Linux Server 8.2

Lydia Parziale

Murthy Garimella

Eric Marins

Filipe Miranda

Paul Novak

Vinodkumar Ramalingam

IBM Z

IBM

Redbooks

International Technical Support Organization

The Virtualization Cookbook for IBM Z Volume 2: Red Hat Enterprise Linux Server 8.2

October 2021

SG24-8303-01

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

Second Edition (October 2021)

This edition applies to Version 7, Release 1of IBM z/VM, and Red Hat Enterprise Linux Servers 8.2,

Contents

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 grant 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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites 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.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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 **ibm.com**/legal/copytrade.shtml

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

FlashCopy®
IBM®
IBM Cloud®
IBM Garage™
IBM Z®
IBM z Systems®
OMEGAMON®
RACF®
Redbooks®
Redbooks (logo)  ®
Satellite™
System z®
Tivoli®
WebSphere®
z Systems®
z/Architecture®
z/OS®
z/VM®

The following terms are trademarks of other companies:

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

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

Ansible, Fedora, OpenShift, Red Hat, are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

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 is Volume 2 of a five-volume series of books entitled *The Virtualization Cookbook for IBM Z®*.

This volume includes the following chapters:

- Chapter 1, “Installing Red Hat Enterprise Linux on LNXADMIN” on page 3, describes how to install and configure Red Hat Enterprise Linux onto the Linux Administration server, which performs the cloning and other tasks.
- Chapter 2, “Automated Red Hat Enterprise Linux installations by using Kickstart” on page 35, describes how to use Red Hat’s `kickstart` tool to create Linux systems. This tool is fundamentally different from cloning in that an automated installation is implemented. You can try `kickstart` and cloning. Understand that these applications attempt to accomplish the same goal of quickly getting Linux systems up and running, and that you do not need to use both.
- Chapter 3, “Working with subscription-manager, yum, and DaNdiFied” on page 45, describes how the Red Hat Network works. It provides centralized management and provisioning for multiple Red Hat Enterprise Linux systems.

`Kickstart` is an easy and fast way to provision your Linux guests in any supported Linux platform. It re-creates the operating system from the beginning by using the `kickstart` profile configuration file that installs the new operating system unattended. It also sets up the new guest according to the definition that was set up in the `kickstart` file.

Usually, Linux is administered by the same team that manages Linux on all platforms. By using `kickstart`, you can create a basic profile that can be used in all supported platforms and customize Linux profiles, as needed.

Cloning requires a better understanding of the z/VM environment and z/VM skills. It is a fast process if you enable the IBM FlashCopy® feature in advance. It clones the disks from a golden image to new disks that are used by the new Linux guest. The process can be automated by using the cloning scripts that are supplied with this book.

It is recommended that you start with *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147 of this series because the IBM® z/VM hypervisor is the foundation (or base “layer”) for installing Linux on IBM Z.

Concept of the series

This book series assumes that you are generally familiar with IBM Z technology and terminology. It does not assume an in-depth understanding of z/VM or Linux. It is written for individuals who want to start quickly with z/VM and Linux, and get virtual servers set up and running in a short time (days, not weeks or months).

Volume 1 starts with a solution orientation, discusses planning and security, and then, describes z/VM installation methods, configuration, hardening, automation, servicing, networking, optional features, and more. It adopts a cookbook-style format that provides a concise, repeatable set of procedures for installing, configuring, administering, and maintaining z/VM. This volume also includes a chapter about monitoring z/VM and the Linux virtual servers that are hosted.

Volumes 2, 3, and 4 assume that you completed all of the steps that are provided in Volume 1. From that common foundation, these volumes describe how to build your own Linux virtual servers on IBM Z hardware under IBM z/VM. The cookbook format continues with installing and customizing Linux.

Volume 5 provides an explanation of the kernel-based virtual machine (KVM) on IBM Z and how it can use the z/Architecture®. It focuses on the planning of the environment and provides installation and configuration definitions that are necessary to build, manage, and monitor a KVM on Z environment. This publication applies to the supported Linux on Z distributions (Red Hat, SUSE, and Ubuntu).

Volumes in this series

This book series consists of the following volumes:

- *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147*,* introduces the concept of the IBM virtualization solution by using z/VM® to run Linux servers on IBM Z. It also describes the z/VM functionalities, planning, installation, and configuration of a two-member SSI with z/VM 7.2.

  For Volume 1, you need at least two IBM Z logical partitions (LPARs) with associated resources and z/VM 7.2 installation media.

- *The Virtualization Cookbook for IBM Z Volume 2: Red Hat Enterprise Linux 8.2 Servers*, SG24-8303*,* describes the installation and customization of Red Hat Enterprise Linux.

  For Volume 2, you need the Red Hat Enterprise Linux Server version 8.2 installation media.

- *The Virtualization Cookbook for IBM z Systems Volume 3: SUSE Linux Enterprise Server 12*, SG24-8890*,* describes the installation and customization of SLES.

  For Volume 3, the SUSE Linux Enterprise Server (SLES) version 12 media.

- *The Virtualization Cookbook for IBM z Systems Volume 4: Ubuntu Server 16.04*, SG24-8354*,* describes the installation and customization of SLES.

  For Volume 4, the initial Ubuntu Server 16.04 LTS media plus resources for mirroring are needed.

- *Virtualization Cookbook for IBM Z Volume 5: KVM* describes the kernel-based virtual machine (KVM) on IBM Z and how it can use the z/Architecture.

  For Volume 5, you need KVM distribution installation media.

Conventions

The following conventions are used in this book:

- Font conventions:
  - **`Monospace and bold`**: Commands that are entered by the user on the command line.
  - `monospace`: Linux file, directories, and commands.
  - `MONOSPACE CAPITALS`: z/VM files, virtual machine and minidisk names, and commands.
  - ***`Monospace bold italics`***: Values that were used to test this book, such as TCP/IP addresses. This font convention is used to signify that you need to replace the ***`example value`*** with the correct value for your system or enterprise.
- Command conventions that are used in this book:
  - z/VM commands are prefixed with **`===>`**
  - z/VM XEDIT subcommands are prefixed with **`====>`**
  - Linux commands that are running as root are prefixed with **`#`**
  - Linux commands that are running as non-root are usually prefixed with **`$`**

Operating system releases that are used in this book

The following releases of operating systems were used in writing this book:

- z/VM Version 7.1.0: GA code, September 2020
- Red Hat Enterprise Linux Version 8: GA code, May 2019

Authors

This book was produced by a team of specialists from around the world working at the IBM International Technical Support Organization (ITSO), Poughkeepsie Center.

Lydia Parziale is a Project Leader for the ITSO team in Poughkeepsie, New York, with domestic and international experience in technology management, including software development, project leadership, and strategic planning. Her areas of expertise include Linux on IBM Z and database management technologies. Lydia is a PMI certified project management professional and a Certified IT Specialist with an MBA in Technology Management. She has been employed by IBM over 25 years in various technology areas.

Murthy Garimella is a Global Solutions Architect for IBM as a part of the Global Strategic Alliance team from Red Hat in Colorado in the US. He has more than 20 years of experience in Information Technology, especially focusing on Cloud, Infrastructure, and Security fields. He holds a masters degree in MBA from University of Colorado and is pursuing his second Master's degree at Harvard University. His areas of expertise include Public Cloud (AWS, Azure, and Oracle), Red Hat products, such as Red Hat Enterprise Linux and Kubernetes (OpenShift Container Platform), CI/ CD, and DevOps.

Eric Marins is a Senior IT Architect in Brazil, focused on hybrid cloud solutions, Infrastructure and Platform solutions, and competencies, including High Availability, Disaster Recovery, Networking, Linux, and Cloud. Eric works with CIO Private Cloud in designing and implementing complex hybrid cloud solutions that involve Infrastructure-as-a-Service (IaaS) and Platform-as-a-Service (PaaS) capabilities. Eric has co-authored several IBM Redbooks publications.

Filipe Miranda is the Global Lead for Red Hat Enterprise Linux for IBM z Systems® and Power Systems for Red Hat Inc. His key responsibility is to help shape the overall Linux on z Systems and Power Systems strategy for Red Hat. With more than 13 years of experience in Linux and Open Source technologies, he joined Red Hat Inc. 9 years ago and is based in Southern California, US. He holds a degree in Data Communication from University of California, Los Angeles, and in Computer Science from the University Paulista in Sao Paulo, Brazil.

Paul Novak is a member of the IBM Washington Systems Center z/VM and Linux (formerly ATS) team in Endicott, New York. Paul has experience in field services, user support, software development, enterprise hosting, enterprise architecture, and client technical solutioning. His areas of expertise are z/VM, Linux, middleware, containers, automation, and web identity and access management (IAM) solutions. Paul previously worked as IBM technical webmaster on a team that was responsible for the world's largest production implementations of IBM WebSphere®, Domino, IBM MQ, and ICS on Linux under z/VM. He is a Senior Certified IT Specialist and holds several degrees. Paul is a multi-generation Endicott IBMer and has more than 25 years of Linux and Open Source technology experience.

Vinodkumar Ramalingam is an Infrastructure Architect in India, working for IBM GTS Offerings and Chief Technology Office. He has over 20 years of experience in Information Technology, specializing in IBM Z Infrastructure. He holds Master's degrees in Computer Science and Business Administration, plus a M.Phil (Computer Science) and Post Graduate Diploma in Cyber Laws. His areas of expertise include IBM Z Security and Modernization. He has written extensively on Mainframe Security and co-authored four IBM Redbooks® on this subject.

Special thanks

Thanks to the following people for their contributions:

Robert Haimowitz, Patrik Hysky
IBM Garage™ for Systems

Bill Lamastro, Tom Ambrosio
IBM CPO

Fred Bader, Bruce Hayden, Richard Lewis, Matthew Mondics, Kenneth Morse, Barry Silliman, Jack Sykes
IBM Washington Systems Center / ATS

Alan Altmark, Marci Beach, Bill Bitner, Kay Blake, Charlie Bryant, Tung-Sing Chong, Michael Donovan, Carol Everitt, John Franciscovich, Glenda Ford, Jacob Gagnon, Leslie Geer, Timothy Greer, John Hollenbeck, Brian Hugenbruch, Emily Hugenbruch, Donald McGlynn, Kathryn Murtha, Damian Osisek, Patricia Rando, Timothy Reynolds, Jon Ruhl, James Switzer, Susan Timashenka, Brian Wade
IBM Endicott

Andreas Bieswanger, Hendrik Brückner, Ingo Franzki, Stephanie Gherghe, Dominik Klein, Steffen Maier, Karl Mayer, Wilhelm Mild, Pradeep Parameshwaran, Eberhard Pasch, Stefan Reimbold, Stefan Raspi
IBM Böblingen

Fernando Aragao Costa,Talita de Souza Machado Miranda
IBM Brazil

Rhette Angus, Eric David Carlson, Clarice Y Robinson, Tausif A Samad, Lucas Sandoval
IBM Shop Z, ordering, and Techline

Chris Mackowski, Jan Stodola, Dan Horak, Filippe Spolti
Red Hat, Inc.

Mike Friesenegger, Mark Post, Ihno Krumreich, Donald Vosberg, and Terry Smith
SUSE Group

Thanks to Michael MacIsaac for the original inception of this cookbook and for his efforts in continually moving the cookbook forward over the years.

Thanks to many others in IBM Endicott, Böblingen, and Poughkeepsie, and to the many who answered questions about the Linux-390 and IBMVM list servers.

Thanks to the authors of the previous editions of this book:

- Authors of the previous IBM Redbooks edition, *The Virtualization Cookbook for IBM Z Volume 2*: Red Hat Enterprise Linux 7.1 Servers, SG24-8303: Lydia Parziale, Berthold Gunreben, Filipe Miranda, Paul W. Novak, and Ken Werner.
- Authors of the previous IBM Redbooks edition, *The Virtualization Cookbook for IBM Z® Volume 1: IBM z/VM 6.3*, SG24-8147: Lydia Parziale, Marian Gasparovic, Berthold Gunreben, Michael MacIsaac, Filipe Miranda, and Daniel Ruutz.
- Authors of the previous IBM Redbooks edition, *z/VM and Linux on IBM System z: The Virtualization Cookbook for SLES 11 SP1*, SG24-7931: Marian Gasparovic and Michael MacIsaac
- Authors of the previous IBM Redbooks edition, *z/VM and Linux on IBM System z: The Virtualization Cookbook for Red Hat Enterprise Linux 6.0*, SG24-7932: Brad Hinson and Michael MacIsaac

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

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 email 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

- Look for us on LinkedIn:

  www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Summary of changes

This summary of changes refers to the series of books. The three volumes are now:

- *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147
- *The Virtualization Cookbook for IBM Z Volume 2: Red Hat Enterprise Linux 8.2 Server*, SG24-8303
- *The Virtualization Cookbook for IBM z Systems Volume 3: SUSE Linux Enterprise Server 12*, SG24-8890

This section describes the technical changes that were made in this edition of the book, previous editions, and other books in the series. This edition might also include minor corrections and editorial changes that are not identified.

Summary of Changes
for SG24-8303-01
for *The Virtualization Cookbook for IBM Z Volume 2: Red Hat Enterprise Linux Server 8.2*
as created or updated on October 19, 2021.

Summary of changes in this book

The following changes were made to this book from the prior publication:

- The z/VM chapters were updated and moved to *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147.
- The SUSE chapters were updated and moved to *The Virtualization Cookbook for IBM z Systems Volume 3: SUSE Linux Enterprise Server 12,* SG24-8890.
- The chapter *Red Hat Network Satellite Server* was removed, mainly because Red Hat Satellite 6.0 Server is no longer available to Red Hat Enterprise Linux on IBM Z, only on the client side. To manage systems, the RH Satellite™ Server will require an x86 system. After RH IBM Satellite Server is online, it can manage all platforms: x86_64, ppc64BE/LE, and s390x.

The following topic is included for the first time:

- Systemd in Linux

Part 1

Red Hat Enterprise Linux 8.2 servers

It is recommended that you start with Volume 1 of this series because IBM z/VM is the base “layer” when you install Linux on IBM Z. Volume 1 starts with an introduction, discusses planning, and then, describes installing z/VM into a two-node single system image (SSI) cluster.

Volume 1 also describes configuration, hardening, automation, and servicing. It adopts a cookbook format that provides a concise, repeatable set of procedures for installing and configuring z/VM by using the SSI clustering feature. After you complete the Volume 1 tasks, you are ready to install a Linux guest by performing the tasks that are described in this book.

This part of the book includes the following topics:

1

Installing Red Hat Enterprise Linux on LNXADMIN

It is time to create the first identity or Multi-Configuration Virtual Machine (MCVM), `LNXADMIN`. An MCVM can be logged on to all members of the single system image (SSI) at the same time. Therefore, it is not possible to migrate an MCVM between SSI members.

This virtual machine serves many administrative purposes, including the following examples:

- Red Hat Enterprise Linux installation server: A file system tree of RPMs and other files that are required for installation are made available with File Transfer Program (FTP). For more information, see 1.2.2, “Copying the Red Hat Enterprise Linux 8.2 installation tree to LNXADMIN” on page 17.
- Red Hat Enterprise Linux Kickstart repository: For hosting the necessary files for automated installations. For more information, see Chapter 2, “Automated Red Hat Enterprise Linux installations by using Kickstart” on page 35.
- Red Hat Enterprise Linux Terminal Server: To easily access any of your Linux virtual machines by using inter-user communication vehicle (IUCV), not TCP/IP. For more information, see 1.2.5, “Configuring IUCV Linux Terminal Server” on page 20.

This chapter includes the following topics:

1.1 Installing the Linux administration system

In this section, we demonstrate how to install Red Hat Enterprise Linux 8.2 on to the identity **LNXADMIN**.

1.1.1 Preparing Red Hat Enterprise Linux 8.2 bootstrap files for LNXADMIN

To IPL a Red Hat Enterprise Linux 8.2 installation system, we used four bootstrap files that must be prepared and copied to the LNXADMIN virtual machine A disk. In this case, the A disk is actually the LNXADMIN directory in the LNX Shared File System (SFS) file pool (LNX:LNXADMIN). Three files will be spooled to the z/VM reader and then punched to IPL Linux:

- The kernel itself, KERNEL.IMG
- A parameter file, GENERIC.PRM
- An initial RAMdisk, INITRD.IMG

A fourth file is the REDHAT.EXEC, which is a small REXX script that is commonly used to clean out the reader, punch the three files, and IPL the reader.

Note: The following list summarizes the installation process:

- Copy the following files: INITRD.IMG, KERNEL.IMG, GENERIC.PRM, and REDHAT.EXEC to the target z/VM user ID by using FTP.
- Edit GENERIC.PRM file and populate it with the correct parameters.
- Start the installation process with REDHAT.EXEC.

Complete the following steps to IPL a Red Hat Enterprise Linux 8.2 installation:

1. Log in as LNXADMIN from the 3270 console. The PROFILE EXEC file runs when you press Enter at the VM READ prompt when you log in. It creates two virtual disks with the SWAPGEN EXEC to use later as swap spaces. This process is shown in Example 1-1.

Example 1-1 PROFILE EXEC process

```
LOGON RDBKRH08
 ICH70001I RDBKRH08 LAST ACCESS AT 13:55:46 ON MONDAY, SEPTEMBER 21, 2020
00: NIC 0600 is created; devices 0600-0602 defined
00: NIC 0620 is created; devices 0620-0622 defined
00: NIC 0640 is created; devices 0640-0642 defined
00: z/VM Version 7 Release 1.0, Service Level 2001 (64-bit),
00: built on IBM Virtualization Technology
00: There is no logmsg data
00: FILES: 0001 RDR,   NO PRT,    NO PUN
00: LOGON AT 11:10:56 EDT WEDNESDAY 09/23/20
z/VM V7.1.0    2020-03013 12:46
```

The PROFILE EXEC also accesses the TCPMAINT 592 disk, which provides access to the TCP/IP tools, as shown in Example 1-2. It also performs other functions, including the automatic IPL of Linux.

Example 1-2 Linking to the TCPMAINT 592 disk

```
DMSVML2060I TCPMAINT 592 linked as 0120 file mode Z
LNXADMIN AT ITSOZVM1 VIA RSCS     2015-04-20 15:29:13 EDT      MONDAY
DIAG swap disk defined at virtual address 300 (64988 4K pages of swap space)
```

```
DIAG swap disk defined at virtual address 301 (129980 4K pages of swap space)
Do you want to IPL Linux from minidisk 100? y/n

===> N
```

Example 1-3 shows how the PROFILE EXEC B information is displayed when it is run.

Example 1-3 PROFILE EXEC B

```
READY;
 profile exec b
592 replaces V (592)
V (592) R/O
212 replaces B (212)
B (212) R/O
SWPSWA0012I DIAG swap disk defined at virtual address 0300 with
SWPSWA0012I 64988 4K pages of swap space.
SWPSWA0012I DIAG swap disk defined at firtual address 0301 with
SWPSWA0012I 129880 4K pages of swap space.
Do you want to IPL Linux from 0100?
```

To view a PROFILE EXEC, run the **TYPE** command, as shown in Example 1-4.

Example 1-4 Viewing the contents of the PROFILE EXEC

```
==> type PROFILE EXEC a
/* PROFILE EXEC FOR LINUX VIRTUAL SERVERS -- MOD 2015-04-10  PWNOVAK  */
/* BOOTING FROM ECKD DASD OR MINIDISKS                                */
/**********************************************************************/
'CP SP CONS CLOSE'                                 /* CLOSE CUR CONLOG */
'CP SP CONS TO LNXADMIN START NAME 'USERID()' CONSLOG'  /* CONSLOG ON */
'CP SET RUN ON'                                    /* RUN DISCONNECTED */
'CP SET PF11 RETRIEVE FORWARD'                     /* RETRIEVE CMD FWD */
'CP SET PF12 RETRIEVE'                             /* RETRIEVE CMD BKW */
'IDENTIFY (ISODATE'                                /* IDENTIFY GUEST   */
'ACCESS 592 T'                                     /* ACCESS TCP TOOLS */
'PIPE CP QUERY' USERID() '| VAR USER'              /* DETERMINE USERID */
 PARSE VALUE USER WITH ID . DSC .                  /* CHECK IF DISCOED */

 IF ( ID <> 'LNXADMIN' ) THEN              /* IF USER IS NOT LNXADMIN */
  DO                                       /* IF USER IS NOT LNXADMIN */
   'ACCESS LNX:LNXADMIN. D'                /* IF USER IS NOT LNXADMIN */
  END                                      /* IF USER IS NOT LNXADMIN */

'SWAPGEN 0300 0524288'             /* MAKE 256M LNXSWAP VDISK AT 0300 */
'SWAPGEN 0301 1048576'             /* MAKE 512M LNXSWAP VDISK AT 0301 */

 IF (DSC = 'DSC') THEN                     /* IF USER IS DISCONNECTED */
  DO
   'CP QUERY TERMINAL'
   'CP QUERY CONSOLE'
   'CP SPOOL CONSOLE STOP'                            /* CONSLOG OFF */
   'CP IPL 0100'                                      /* BOOT LINUX  */
  END
 ELSE                                 /* USER IS INTERACTIVE SO PROMPT */
  DO
```

```
  SAY 'DO YOU WANT TO IPL LINUX FROM MINIDISK 100? Y/N'
  PARSE UPPER PULL ANSWER .
  IF (ANSWER = 'Y') THEN
   DO
    'CP SPOOL CONSOLE STOP'                          /* CONSLOG OFF */
    'CP IPL 0100'                                    /* BOOT LINUX  */
   END
 END
```

For example, in the case of Red Hat Enterprise Linux 8.2 on an IBM z/VM environment, the PROFILE EXEC might look as shown in Figure 1-1 (it can be different in your case based on your REXX profile).

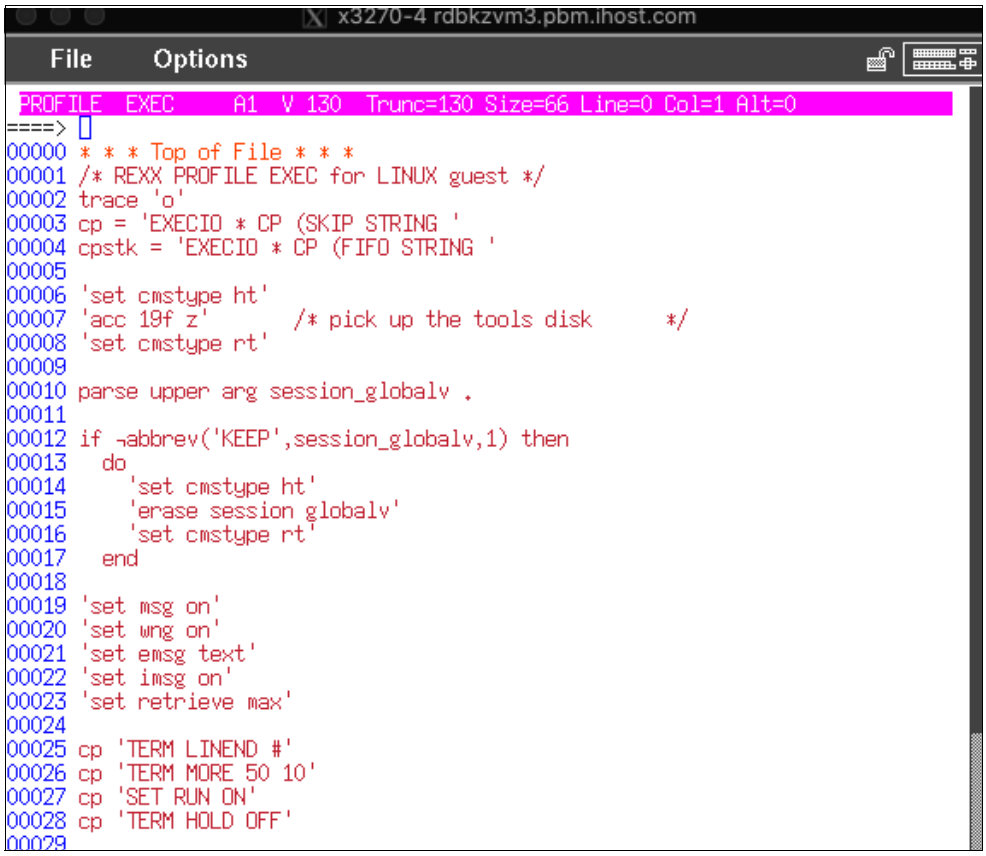

Figure 1-1 PROFILE EXEC A

2. Verify that the LNXADMIN directory in the LNX Shared File System (SFS) pool (`LNX:LNXADMIM`) is accessed as file mode A by running the **`q accessed`** command to view the available directories for Red Hat Enterprise Linux 8.2, as shown in Figure 1-2. You can see that Red Hat Enterprise Linux 8.2 is available in Mode B, which includes the install software.

```
 q accessed
Mode  Stat     Files  Vdev  Label/Directory
A       R/W        4  191   RH7191
B       R/O       32  212   RHEL82
C       R/W        1  300   LXSWAP
D       R/W        1  301   LXSWAP
S       R/O      696  190   MNT190
V       R/O      624  592   TCM592
Y/S     R/O     1120  19E   MNT19E
Z       R/O     8330  19F   MNT19F
Ready;
```

Figure 1-2 Query to list the directories

3. Transfer the bootstrap files (`exec` and `img`) from the external FTP server to your local A disk (see Example 1-5).

Example 1-5 Transferring the bootstrap files

```
==> ftp 9.60.87.87
VM TCP/IP FTP Level 630
Connecting to 9.60.87.87, port 21
220 (vsFTPd 3.0.2)
USER (identify yourself to the host):
lydiap
>>>USER lydiap
331 Please specify the password.
Password:

>>>PASS ********
230 Login successful.
Command:

cd /linux/rhel82/images
get redhat.exec redhat.exec
locsite fix 80
binary
get kernel.img kernel.img
get initrd.img initrd.img
```

4. After all files are copied, edit the `generic.prm` file with the parameters:

```
==> listfiles * * a
INITRD   IMG      A1
KERNEL   IMG      A1
PROFILE  EXEC     A1
REDHAT   EXEC     A1
SWAPGEN  EXEC     A1
```

```
PROFILE  XEDIT    A1
Ready; T=0.01/0.01 12:23:26
```

5. Edit the **REDHAT EXEC** file to the following format:

```
==> xedit REDHAT EXEC a
```

> **Note:** If you are unfamiliar with **xedit**, see “XEDIT cheat sheet” on page 230.

```
/* */
'CL RDR'
'PURGE RDR ALL'
'SPOOL PUNCH * RDR'
'PUNCH KERNEL  IMG * (NOH'
'PUNCH INITRD  IMG * (NOH'
'CH RDR ALL KEEP NOHOLD'
'I 00C'
```

6. Edit a new file that is called `generic prm` by adding the correct parameters and system information:

```
==> xedit GENERIC PRM a

ro ramdisk_size=40000 cio_ignore=all,!condev
ip=9.12.7.96::9.12.4.1:20:vmlnx2-1.itso.ibm.com:enccw0.0.0600:none
rd.znet=qeth,0.0.0600,0.0.0601,0.0.0602,layer2=1
nameserver=9.12.x.7 nameserver=9.12.x.6
inst.repo=ftp://ftproot:myftppassword@x.xx.xx.87/ftp/linux/rhel82
rd.dasd=0.0.0100 rd.dasd=0.0.0200
vnc vncpassword=12345678
```

where `ip` your IP address.

1.1.2 Installing Red Hat Enterprise Linux 8.2

Complete the following steps to begin the Linux installation:

1. To begin the installation program, run the `REDHAT EXEC`. Many windows of output scroll by, as shown in Example 1-6.

```
==> redhat82 rh07
```

Example 1-6 Example output from REDHAT EXEC

```
Starting Terminate Plymouth Boot Screen...
Starting pre-anaconda logging service...
Starting Anaconda NetworkManager configuration...
Starting Hold until boot process finishes up...
Starting OpenSSH ed25519 Server Key Generation...
Starting Service enabling compressing RAM with zRam...
Starting OpenSSH rsa Server Key Generation...
Starting Login Service...
Starting OpenSSH ecdsa Server Key Generation...
Starting Hardware RNG Entropy Gatherer Wake threshold service...
[ [0;32m  OK   [0m] Started Daily Cleanup of Temporary Directories.
[ [0;32m  OK   [0m] Reached target Timers.
[   59.866175] anaconda[1930]: Starting installer, one moment...
```

```
[   59.866283] anaconda[1930]: 16:16:00 Please ssh install@rdbkrh07.pbm.ihost.com
(129.40.23.138) to begin the install.
```

2. After you log on by using a Secure Shell (SSH) session with the install user, the Virtual Network Computing (VNC) server is enabled according to the parameters that were defined in the `GENERIC.PRM` file. The messages that is shown in Example 1-7 display installer progress.

Example 1-7 Starting the installer

```
Starting installer, one moment...
anaconda 29.19.2.17-1.el8 for Red Hat Enterprise Linux 8.2 started.
 * installation log files are stored in /tmp during the installation
 * shell is available in second TMUX pane (Ctrl+b, then press 2)
 * if the graphical installation interface fails to start, try again with the
   inst.text bootoption to start text installation
 * when reporting a bug add logs from /tmp as separate text/plain attachments
18:26:16 Starting VNC...
18:26:17 The VNC server is now running.
18:26:17

You chose to execute vnc with a password.

18:26:17 Please manually connect your vnc client to vmlnx2-1.itso.ibm.com:1
(9.12.7.96:1) to begin the install.
```

3. Start a VNC client (for example, RealVNC). Connect to the server with your IP address or your fully qualified domain name with a `:1` appended to the end, as shown in Figure 1-3.

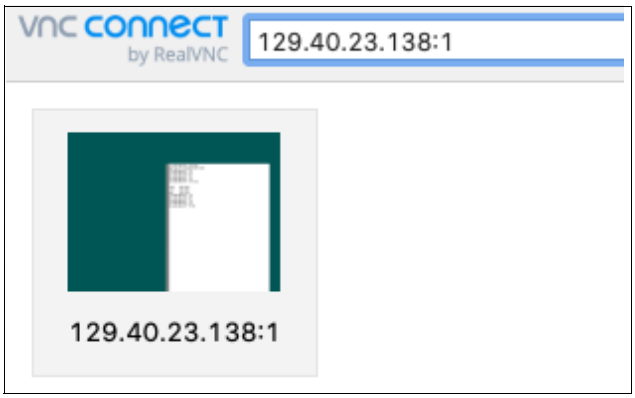

Figure 1-3 Connecting with VNC client

4. When you are prompted for a password, enter the password that is specified in the `LNXADMIN GENERIC.PRM` file (12345678). In the following example, Linux is installed with the Domain Name Server (DNS) name, `rdbkrh07.pbm.ihost.com`.

1.1.3 Stage 2 of the Red Hat Enterprise Linux 8.2 installation

After you connect by using VNC, complete the following steps:

1. Use the first window to select the language that are to be used by the installer. Select **English** and then, click **Continue** (see Figure 1-4).

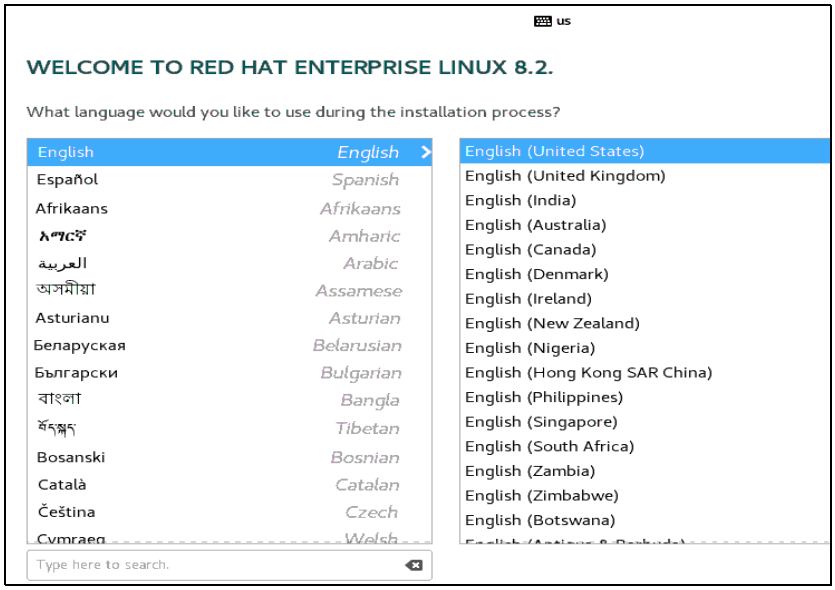

Figure 1-4 Select the installer language

2. Red Hat completely redesigned the Red Hat Enterprise Linux 8 main Installation Summary window, as shown in Figure 1-5. Click **Installation Destination** to set up the disks to use in this step and to set up disk partitioning.

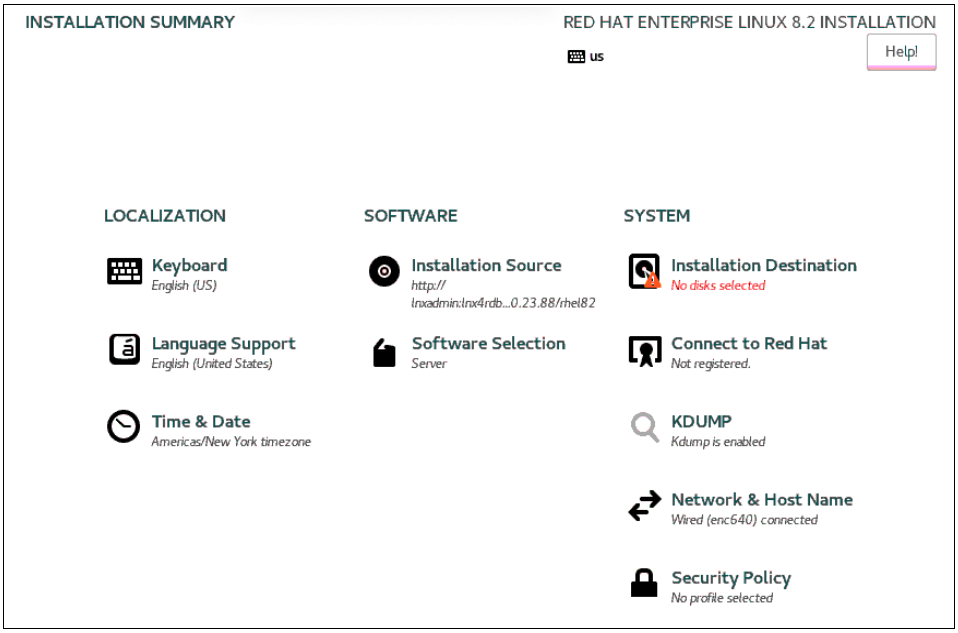

Figure 1-5 Main Installation Summary window

3. Select the **dasda** disk from the Local Standard Disks menu (as shown in Figure 1-6) and select **Automatically configure partitioning**. You configure the other DASD, `dasdb`, after the installation. Click **Done**.

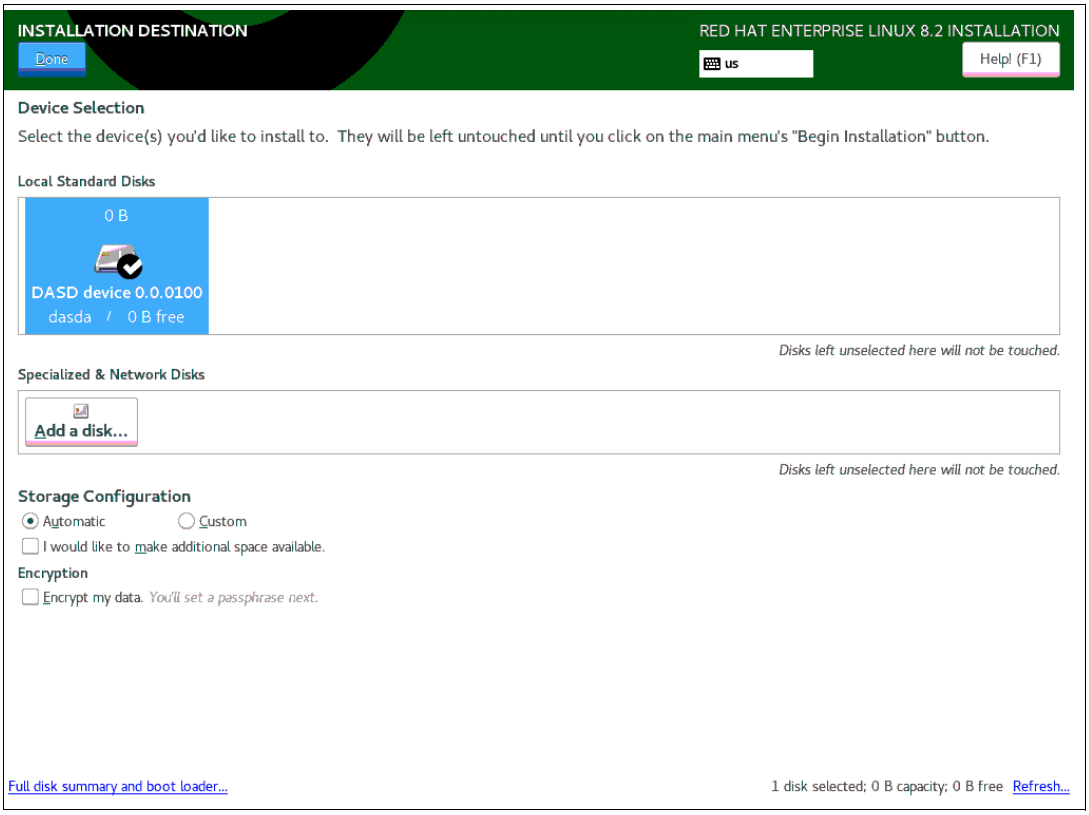

Figure 1-6 Disk selection

4. The unformatted DASD disks are low-level formatted by using **`dasdfmt`** from the graphical installer automatically, as shown in Figure 1-7. You do not need to format unformatted DASD from a Linux terminal, which was required in previous versions of Red Hat Enterprise Linux.

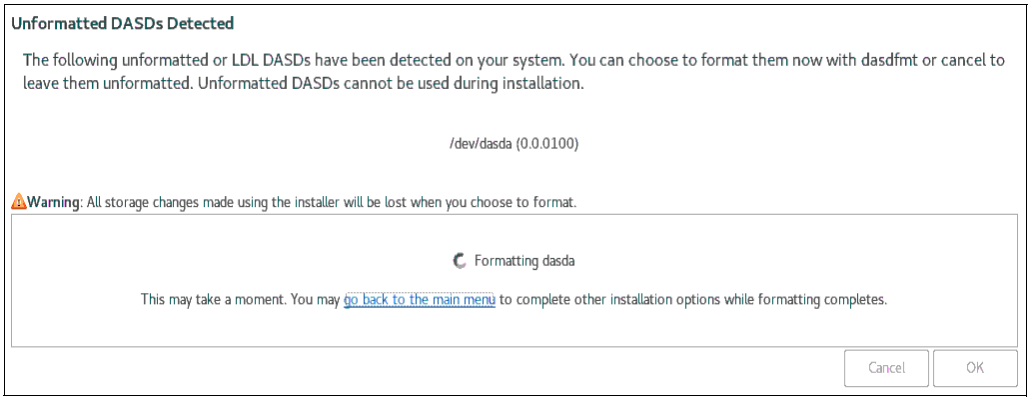

Figure 1-7 Installer dasdfmt

5. Keep the defaults for all other options. Most of the options were preconfigured by the `GENERIC.PRM` file. Click **Begin Installation**.

6. The last phase of the installer presents a window (see Figure 1-8) that shows the installation progress and provides the option to define the user **root** password and to add other users. At a minimum, define a secure **root** password. After the installation process completes, click **Reboot**.

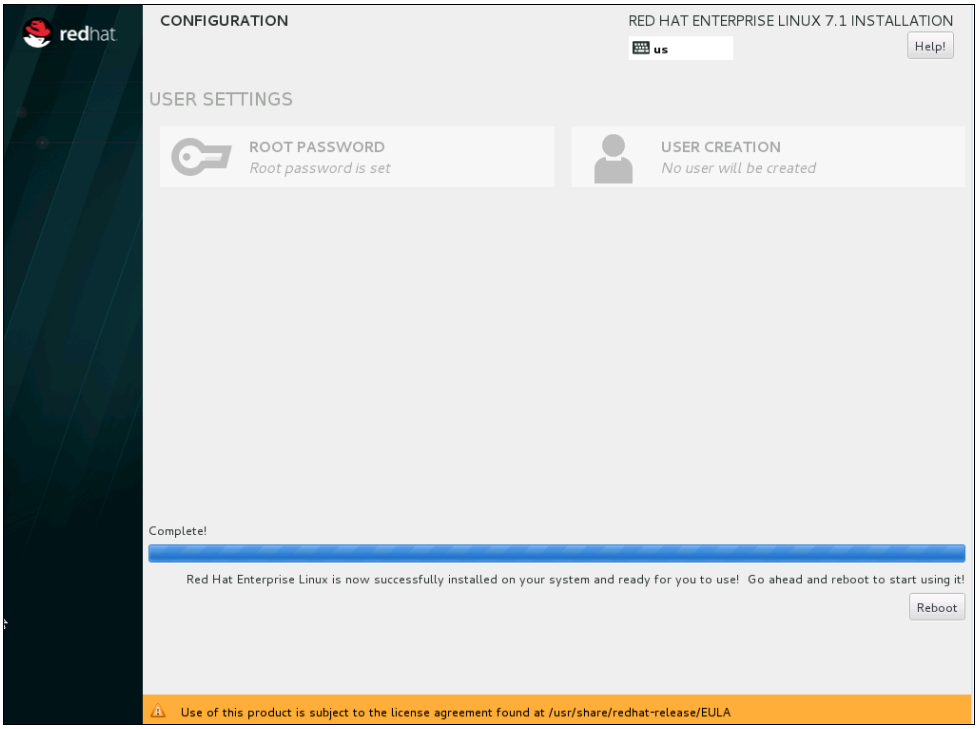

Figure 1-8 Defining the user root password and reboot after the installation completes

1.1.4 Booting your new Linux system from disk

A default system is now installed onto minidisk 100. Return to your z/VM 3270 session and your newly installed system automatically IPLs again. Your system continues to boot until a log in prompt is presented.

After logging in, complete these steps:

1. Disconnect from the 3270 session:

   ```
   ==> #cp disc
   ```

2. Start an SSH session to the new Linux administration system by logging in as the **root** user:

   ```
   login as: root
   root@rdbkrh07.pbm.ihost.com's password:
   ```

3. Use the following command to verify the name of the kernel, network node host name, kernel version number, and release level, kernel release date, machine hardware name, CPU type, hardware platform and operating system name:

   ```
   # uname -a
   ```

   You see results that are similar to the results that are shown in Example 1-8 on page 13.

Example 1-8 System information

```
rdbkrh07 -]# uname -a
rdbkrh07.pbm.ihost.com 4.18.0-193-el8.s390x #1 SMP Fri Mar27 14:43:09 UTC
s390x s390x s390x GNU/Linux
rdbkrh07 -]#
```

4. Run the following command to verify the Red Hat Enterprise Linux version:

   # cat /etc/redhat-release

   You see results that are similar to the results that are shown in Example 1-9.

Example 1-9 Release information

```
rdbkrh07 -]# cat /etc/redhat-release
Red Hat Enterprise Linux release 8.2(Ootpa)
```

1.1.5 Setting up the data DASD disk after the installation process

In this section, we describe how to add a DASD disk (the data disk) to your LNXADMIN guest. This task demonstrates how to add DASD disks to a running Red Hat Enterprise Linux guest.

This task also shows the easy of migrating data to another guest, if necessary, because your data is on a separate volume. Because the data is on a separate volume, the data volume group (VG) can span multiple disks when it uses the Logical Volume Manager (LVM); therefore, easily increasing the logical volume (LV) without guest disruption.

Complete the following steps:

1. Use the **lsdasd** command to check the DASD disks that are active in your system:

   ```
   # lsdasd
   Bus-ID     Status      Name      Device  Type  BlkSz  Size      Blocks
   ==============================================================================
   0.0.0100   active      dasda     94:0    ECKD  4096   7042MB    1802880
   ```

2. Remove the device from the system blacklist by using the command **cio_ignore**:

   ```
   # cio_ignore -r 0.0.0200
   ```

3. Enable the minidisk 200 with the **chccwdev** command:

   ```
   # chccwdev -e 200
   Setting device 0.0.0200 online
   Done
   ```

4. Edit the /etc/dasd.conf to change it to persistent:

   ```
   # echo '0.0.0200' >> /etc/dasd.conf
   ```

5. Use the **lsdasd** command to verify that minidisk 200 is enabled:

   ```
   # lsdasd
   Bus-ID     Status      Name      Device  Type  BlkSz  Size      Blocks
   ==============================================================================
   0.0.0100   active      dasda     94:0    ECKD  4096   7042MB    1802880
   0.0.0200   active      dasdb     94:4    ECKD  4096   7042MB    1802880
   ```

6. To format the minidisk, use the **dasdfmt** command:

   ```
   # dasdfmt -b 4096 -y -f /dev/dasdb
   Finished formatting the device.
   Rereading the partition table... ok
   ```

7. Create a partition for your DASD device by using the **fdasd** command:

```
# fdasd -a /dev/dasdb
reading volume label ..: VOL1
reading vtoc ..........: ok
auto-creating one partition for the whole disk...
writing volume label...
writing VTOC...
rereading partition table...
```

8. Verify all of your partitions:

```
# cat /proc/partitions
major minor  #blocks  name

  94        0    7211520 dasda
  94        1     512016 dasda1
  94        2    6699408 dasda2
 253        0     720896 dm-0
 253        1    5931008 dm-1
  94        4    7211520 dasdb
  94        5    7211424 dasdb1
```

9. To create a re-sizable file system for the data disk, you set it up as an LVM. The following steps demonstrate how to create an LVM logical partition (LPAR) on a new volume group by using the disk `dasdb1`:

   a. The **pvcreate** command initializes partitions for use by LVM:

```
# pvcreate /dev/dasdb1
Physical volume "/dev/dasdb1" successfully created
```

   b. Use the **vgcreate** command to create a volume group:

```
# vgcreate vgdata /dev/dasdb1
Volume group "vgdata" successfully created
```

   c. To gather more information about the volume group, use the **vgdisplay** command:

```
# vgdisplay vgdata

--- Volume group ---
  VG Name               vgdata
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               6.88 GiB
  PE Size               4.00 MiB
  Total PE              1760
  Alloc PE / Size       0 / 0
  Free  PE / Size       1760 / 6.88 GiB
  VG UUID               V8dFs4-3zK7-cPzr-O5OC-dPHQ-nUeI-YTVmVQ
```

d. The **lvcreate** command creates a logical volume:

```
# lvcreate -l 1760 -n lvdata vgdata
Logical volume "lvdata" created
```

e. Use the **lvdisplay** command to gather more details about the new logical volume:

```
# lvdisplay vgdata
--- Logical volume ---
  LV Path                /dev/vgdata/lvdata
  LV Name                lvdata
  VG Name                vgdata
  LV UUID                Fuf1eA-UcvP-5vB2-JcQY-8zBx-aFht-okbsVM
  LV Write Access        read/write
  LV Creation host, time vmlnx2-1.itso.ibm.com, 2015-04-14 14:25:01 -0400
  LV Status              available
  # open                 0
  LV Size                6.88 GiB
  Current LE             1760
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     1024
  Block device           253:2
```

10. Create a file system for the new logical volume:

```
# mkfs.xfs /dev/vgdata/lvdata

meta-data=/dev/vgdata/lvdata     isize=256    agcount=4, agsize=450560 blks
         =                       sectsz=4096  attr=2, projid32bit=1
         =                       crc=0        finobt=0
data     =                       bsize=4096   blocks=1802240, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=0
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=4096  sunit=1 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

11. Create a directory to serve as a mount point for the new XFS file system:

```
# mkdir -p /var/ftp/pub
```

12. Edit the `/etc/fstab` file to permanently add a mount point for the new XFS file system:

```
# echo "/dev/vgdata/lvdata  /var/ftp/pub  xfs defaults 0 0" >> /etc/fstab
```

13. To reload the `/etc/fstab` configuration, use the **mount** command:

```
# mount -a
```

14. Check that all file systems are mounted with the **mount** command:

```
# mount
....
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,seclabel)
/dev/dasda1 on /boot type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
/dev/mapper/vgdata-lvdata on /var/ftp/pub type xfs
(rw,relatime,seclabel,attr2,inode64,noquota)
```

The installation of Red Hat Enterprise Linux 8.2 on the Linux administration system is now complete.

1.2 Configuring the Linux administration system

Now that your Linux administration system is installed, it must be configured. The following steps are involved:

1. Enable swap on virtual disks (VDisks).
2. Copy the Red Hat Enterprise Linux 8.2 installation tree to LNXADMIN.
3. Configure the yum DVD repository.
4. Configure vsftpd.
5. Configure IUCV Linux Terminal Server.
6. Copy the files that are associated with this book.
7. Reboot the system.

These steps are described next.

1.2.1 Enabling swap on virtual disks

Each time that the LNXADMIN runs the common `PROFILE EXEC`, two virtual disks (VDisks) are created at virtual addresses `300` and `301`. On the Linux guest, these in-memory disks can be used as swap devices. This section explains how to use these disks as swap devices.

Important: To use virtual disks (VDisk) for swap partitions, you must enable them manually and not during the installation process because Red Hat Enterprise Linux 8.x refers to these disks by their Universally Unique Identifiers (UUIDs) by default.

Because the UUIDs of the VDisks change every time that the `PROFILE EXEC` of the Linux guest is loaded, new UUIDs are generated, which makes it unable to access the swap disks. This section shows how to change the default behavior of Red Hat Enterprise Linux 8.x to access the swap disks on VDisks by their paths rather than their UUIDs.

Dracut manages the initramfs for disks and partitions management. The Persistent Policy option is used to address the disks and partitions. A `<policy>` can be any directory name (`ex- by-uuid`, `by-label`, and so on) Dracut options, such as `f` overwrite the existing `initramfs` file. A command, such as **`zipl`**, is part of the IBM Z initial program load (IPL) and is a loader with configuration files.

Dracut creates an initial image that is used by the kernel for pre-loading the block device modules (such as IDE, SCSI, or RAID), which are needed to access the root file system, mounting the root file system, and booting into the real system.

At boot time, the kernel unpacks that archive into RAM disk, and mounts and uses it as initial root file system. All finding of the root device occurs in this early user space.

To enable swap on the virtual disks, complete the following steps:

1. Change the `/etc/dracut.conf` file by adding the following persistent policy:

   ```
   # echo 'persistent_policy=by-path' >> /etc/dracut.conf
   ```

2. The following command updates the initramfs image and updates the bootloader. Because the newly generated initramfs image is the same as before, you will not need to update the zipl configuration:

   ```
   # dracut -f
   If the initramfs image already exists, dracut will display an error message,
   and to overwrite the existing image, you have to use the ‘f’ or ‘--force’
   option.
   ```

```
# zipl -V
If zipl is called without  parameters,  it  searches  the configuration file
for a section specifying the default action. The default configuration file is
available at ‘/etc/zipl.conf’
```

3. Activate the VDisks:

   a. Use the `cio_ignore` utility to remove the DASD from the list of ignored devices:

   ```
   # cio_ignore -r 0.0.0300
   # cio_ignore -r 0.0.0301
   ```

   b. Set the devices online:

   ```
   # chccwdev -e  0.0.0300
   Setting device 0.0.0300 online
   Done

   # chccwdev -e  0.0.0301
   Setting device 0.0.0301 online
   Done
   ```

4. **Edit the `/etc/dasd.conf` file to change it to persistent:**

   ```
   # echo '0.0.0300' >> /etc/dasd.conf
   # echo '0.0.0301' >> /etc/dasd.conf
   ```

5. View the DASDs that are online by using the `lsdasd` command-line utility:

   ```
   # lsdasd
   Bus-ID     Status      Name      Device  Type  BlkSz  Size      Blocks
   ==============================================================================
   0.0.0100   active      dasda     94:0    ECKD  4096   7042MB    1802880
   0.0.0200   active      dasdb     94:4    ECKD  4096   7042MB    1802880
   0.0.0300   active      dasdc     94:8    FBA   512    256MB     524288
   0.0.0301   active      dasdd     94:12   FBA   512    512MB     1048576
   ```

6. Activate the swap partition on that disk by using the `swapon` command-line utility:

   ```
   # swapon -p 5 /dev/disk/by-path/ccw-0.0.0300-part1
   # swapon -p 4 /dev/disk/by-path/ccw-0.0.0301-part1
   ```

7. Check the activated swap devices:

   ```
   # swapon --show
   NAME        TYPE        SIZE USED PRIO
   /dev/dm-0   partition   704M   0B   -1
   /dev/dasdc1 partition 253.9M   0B    5
   /dev/dasdd1 partition 507.8M   0B    4
   ```

8. Add the swap disks to the `/etc/fstab`:

   ```
   # echo '/dev/disk/by-path/ccw-0.0.0300-part1 swap swap pri=5 0 0' >> /etc/fstab
   # echo '/dev/disk/by-path/ccw-0.0.0301-part1 swap swap pri=4 0 0' >> /etc/fstab
   ```

1.2.2 Copying the Red Hat Enterprise Linux 8.2 installation tree to LNXADMIN

Copy the Red Hat Enterprise Linux 8.2 DVD ISO, with the other files that are associated with this book from the external FTP server to the Linux administration system by using the **`curl`** command:

```
# cd /var/ftp/pub
# curl http://lnxadmin:lnx4rdbk@rdbkftp1.pbm.ihost.com//RHEL-00404.0-390x
-dvd1.iso -o RHEL-8.2.0-20200404.0-s390x-dvd1.iso
```

```
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 5495M  100 5495M    0     0  528M      0  0:00:10 0:00:10 --:--:-- 283M
```

This command takes approximately 2 - 5 minutes to complete.

1.2.3 Configuring the yum DVD repository

Complete the following steps to configure the yum DVD repository:

1. Create a directory to hold the installation tree by using the **mkdir** command and mount loop the DVD ISO into that directory by using the **mount** command:

   ```
   # mkdir rhel82
   # mount -o loop RHEL-8.2-Server-s390x.iso rhel82/
   mount: /dev/loop0 is write-protected, mounting read-only
   ```

2. Create the /etc/yum.repos.d/dvd.repo file with the following content to set up a local yum repository to use to install more packages from the DVD installation tree:

   ```
   [DVD]
   name= RHEL8.2 DVD ISO
   baseurl=file:///var/ftp/pub/rhel82/
   enable=1
   gpgcheck=1
   ```

3. Install the Red Hat GNU Privacy Guard (GPG) key:

   ```
   # rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
   ```

4. Verify the new DVD repository:

   ```
   # yum update
   Loaded plugins: product-id, subscription-manager
   This system is not registered to Red Hat Subscription Management. You can use
   subscription-manager to register.
   DVD  | 4.1 kB  00:00:00
   (1/2): DVD/group_gz | 112 kB  00:00:00
   (2/2): DVD/primary_db | 2.8 MB  00:00:00
   ```

   Note: Red Hat signs each RPM with a private GPG key, which is compared to your public key. Whenever a package is installed, the package manager verifies the package signature. This method ensures that the RPM is a genuine, unaltered package.

   To check an RPM signature, use the following command:

   ```
   # rpm -K package_name.s390x.rpm
   package_name.s390x.rpm: rsa sha1 (md5) pgp md5 OK
   ```

You are now ready to use **yum** to install or upgrade an RPM package. To install a package, use **yum -y install <*packagename*>**. Yum installs the specified packages and automatically resolves dependencies for you.

Note: You must not specify the package version on the command line; only the package name.

1.2.4 Configuring vsftpd

Complete steps to configure vsftpd:

1. Install vsftpd from the local yum repository:

```
# yum -y install vsftpd
Loaded plugins: product-id, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use
subscription-manager to register.
Resolving Dependencies
--> Running transaction check
---> Package vsftpd.s390x 0:3.0.2-9.el7 will be installed
--> Finished Dependency Resolution
...
Installed:
  vsftpd.s390x 0:3.0.2-9.el7
Complete!
```

2. Edit the `/etc/vsftpd/vsftpd.conf` file and uncomment the following lines:

```
ascii_upload_enable=YES
ascii_download_enable=YES
```

3. Start the `vfstpd` service and check the status of the `vfstpd` service by using the `systemctl` command:

```
# systemctl start vsftpd.service
# systemctl status vsftpd.service
vsftpd.service - Vsftpd ftp daemon
   Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; enabled; vendor
pres>
   Active: active (running) since Wed 2020-09-23 16:04:20 EDT; 18min ago
  Process: 1136 ExecStart=/usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
(code=exited>
 Main PID: 1147 (vsftpd)
    Tasks: 1 (limit: 101854)
   Memory: 856.0K
   CGroup: /system.slice/vsftpd.service
           ··1147 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf

Sep 23 16:04:20 rdbkrh07.pbm.ihost.com systemd[1]: Starting Vsftpd ftp
daemon...
Sep 23 16:04:20 rdbkrh07.pbm.ihost.com systemd[1]: Started Vsftpd ftp daemon.
```

4. Enable the `vsftpd` service permanently by using the `systemctl` command:

```
# systemctl enable vsftpd.service
ln -s '/usr/lib/systemd/system/vsftpd.service'
'/etc/systemd/system/multi-user.target.wants/vsftpd.service'
```

5. Enable the `ip_conntrack_ftp` kernel module:

```
# modprobe -i ip_conntrack_ftp
```

6. To permanently enable the ip_conntrack_ftp, create the `/etc/sysconfig/modules/iptables.modules` file with the following content to enable the kernel module at IPL:

```
#!/bin/sh
exec /sbin/modprobe ip_conntrack_ftp >/dev/null 2>&1
```

Note: The number sign (#) on the first line must be included in your file.

7. Configure iptables to allow connections on the ftp port number:

```
firewall-cmd --permanent --zone=public --add-port=21/tcp
firewall-cmd --reload
```

8. Configure SElinux boolean to allow the looped mounted DVD ISO to be accessed:

```
# yum install -y policycoreutils-2.2.5-15.el7.s390x
# setsebool -P ftpd_full_access on
```

The Red Hat Enterprise Linux 8.2 installation tree is ready to be shared over FTP on the `LNXADMIN` system under the `/var/ftp/pub` directory.

1.2.5 Configuring IUCV Linux Terminal Server

A Linux Terminal Server allows access to the console without a functioning TCP/IP stack in z/VM.

To set up the Linux Terminal Server, follow the instructions for z/VM as described in 11.5, “Setting up the IUCV Linux Terminal Server” on page 221, and on Linux guests.

For the Red Hat Enterprise Linux Terminal Server, you configure IUCV to establish terminal sessions to target Red Hat Enterprise Linux systems by using hypervisor console (HVC) terminal devices. Complete the following steps:

1. Add a ts-shell group to access the ts-shell:

```
# groupadd tsgroup
```

2. Add a user to access the ts-shell:

```
# useradd tsuser1 -m -s /usr/bin/ts-shell -g ts-shell -G tsgroup
```

3. Grant authorization to the ts-shell users by editing the `/etc/iucvterm/ts-authorization.conf` file:

```
# echo '@tsgroup = list:LINUX1,LINUX2' >> /etc/iucvterm/ts-authorization.conf
```

4. Connect to the terminal server by using SSH and by using `tsuser1`:

```
# ssh tsuser1@vmlnx2-1.itso.ibm.com
Last login: Thu Apr 16 11:38:54 2015 from 9.12.5.134
Welcome to the Terminal Server shell.
Type 'help' to get a list of available commands.
tsuser1@ts-shell>
```

5. You can now list the systems that are allowed to be accessed by using IUCV and by using the **`list`** command:

```
tsuser1@ts-shell> list
LINUX1
LINUX2
```

6. To connect to the Linux system that you want, use the **`connect`** command:

```
tsuser1@ts-shell> connect LINUX1
ts-shell: Connecting to LINUX1 (terminal identifier: lnxhvc0)...
iucvconn: The target z/VM guest virtual machine is not logged on: Network is
unreachable
ts-shell: Connection ended
```

The connect command failed because no Linux systems are running yet. For more information about installing LINUX1 and LINUX2, see Chapter 2, “Automated Red Hat Enterprise Linux installations by using Kickstart” on page 35.

The required changes to the Red Hat Enterprise Linux target systems are specified in the Kickstart file as described in 1.2.6, “Configuring Kickstart” on page 21.

After you create Linux1 and Linux2, see , “For more information about Kickstart, see this web page.” on page 36, and 2.3, “Configuring LINUX2 for Kickstart by using Fibre Channel Protocol devices” on page 38. Repeat step 6 in the previous procedure now.

Connect to the Linux system that you want by using the **connect** command:

```
tsuser1@ts-shell> connect LINUX1
   ts-shell: Connecting to Linux1 (terminal identifier: lnxhvc0)...
```

Press **Enter**.

```
   Red Hat Enterprise Linux Server 7.1 (Maipo)
   Kernel 3.10.0-229.el7.s390x on an s390x

   linux1 login:
```

Now, log on to LINUX1 by using IUCV.

Note: By default, when you use Red Hat Enterprise Linux, the terminal TERM variable of all of the serial connections are set to dumb. Log in by using the ts-shell, and then, change this setting for the terminal type to the type that suits your needs:

```
export TERM=xterm
```

The changes take effect immediately.

If you are connecting to the Linux Terminal Server by using PuTTy, do not forget to set the correct terminal type in the PuTTY configuration: Terminal -> Keyboard, and then, set the terminal type.

1.2.6 Configuring Kickstart

The installer generates a Kickstart file at the end of every installation. The Kickstart file is based on the answers that are provided during the interactive installation. This Kickstart file is named anaconda-ks.cfg, and it is in the /root/ directory. This file is used as a template for LINUX1.

For more information about the Kickstart commands and options, see this web page.

Complete the following steps to set up the Kickstart directory that is shared by the FTP server:

1. Run the following command to create the Kickstart directory on /var/ftp/pub:

   ```
   # mkdir /var/ftp/pub/kickstart
   ```

2. Copy the /root/anaconda-ks.cfg Kickstart template file to /var/ftp/pub/kickstart/linux1-ks.cfg and grant the file the correct permissions by using the following command:

   ```
   # cp /root/anaconda-ks.cfg /var/ftp/pub/kickstart/linux1-ks.cfg
   # chmod +r linux1-ks.cfg
   ```

3. Red Hat Enterprise Linux is installed on `LINUX1` by using an automated process that uses Kickstart. `LINUX1` uses emulated DASD (EDEV) for the Red Hat Enterprise Linux installation. EDEVs are configured on Linux in the same way that DASD is configured. Edit the Kickstart file that is named `linux1-ks.cfg`, as shown in Example 1-10.

Example 1-10 Edit the Kickstart file

```
#version=RHEL8
# System authorization information
auth --enableshadow --passalgo=sha512

# IBM REDBOOKS RHEL8 TEMPLATE KICKSTART FOR DASD
# Use network installation
install
url --url="ftp://9.12.7.96/pub/rhel82"
# Use text mode install
text
ignoredisk --only-use=dasda
# Keyboard layouts
keyboard --vckeymap=us --xlayouts='us'
# System language
lang en_US.UTF-8
# Network information
network  --bootproto=static --device=enccw0.0.0600 --gateway=9.12.4.1
--ip=9.12.7.98 --nameserver=9.12.6.6,9.12.6.7 --netmask=255.255.240.0 --noipv6
--activate  --hostname=linux1.itso.ibm.com
# Root password
rootpw --iscrypted
$6$pr46QGx7PLwzthjk$41E7GLPSsD//jHPwbQc7/CAG2SSQSkGg/pcveQUXz2IIVLOLCXH2So8n.e1
rFMjqLrfMYWifE7qY2NFfygedw/
# System timezone
timezone America/New_York
# Skip X
skipx
# System bootloader configuration
bootloader --location=mbr --append="hvc_iucv=8 console=hvc0 console=ttyS0"
zerombr
# Partition clearing information
clearpart --all
autopart --type=lvm
reboot

%packages
@core
kexec-tools

%end

%post --log=/root/post.log

# Enable the DVD repo
cat > /etc/yum.repos.d/dvd.repo <<EOF
[DVD]
name= RHEL8.2 DVD ISO
baseurl=ftp://9.12.7.96/pub/rhel82/
enable=1
```

```
gpgcheck=1
EOF

#import RedHat GPG key to verify packages authenticity during yum package
install
rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release

# Enable the VDISKs for swap
echo 'persistent_policy=by-path' >> /etc/dracut.conf
dracut -f
zipl
cio_ignore -r 0.0.0300
cio_ignore -r 0.0.0301
chccwdev -e  0.0.0300
chccwdev -e  0.0.0301
echo '0.0.0300' >> /etc/dasd.conf
echo '0.0.0301' >> /etc/dasd.conf
echo '/dev/disk/by-path/ccw-0.0.0300-part1 swap swap pri=5 0 0' >> /etc/fstab
echo '/dev/disk/by-path/ccw-0.0.0301-part1 swap swap pri=4 0 0' >> /etc/fstab

# Detach CMS DASD disks for z/VM SSI LGR
cat > /etc/rc.d/rc.local <<\EOF
#!/bin/bash
# Detach CMS DASD disks for z/VM SSI LGR
for d in 0190 019D 019E 0592; do
  vmcp q v $d &> /dev/null && echo -n "z/VM disk " && vmcp detach $d
done
exit 0
EOF
chmod +x /etc/rc.d/rc.local

#Enable IUCV hvc0 for the Linux system
ln -s /etc/systemd/system/serial-getty\@hvc0.service
/lib/systemd/system/serial-getty\@.service

%end

%addon com_redhat_kdump --enable --reserve-mb='4096'

%end
```

4. Create another Kickstart file that is named `linux2-ks.cfg` by using the following command and grant it the correct permissions:

   ```
   # cp /var/ftp/pub/kickstart/linux1-ks.cfg /var/ftp/pub/kickstart/linux2-ks.cfg
   # chmod +r linux2-ks.cfg
   ```

5. Red Hat Enterprise Linux is installed on `LINUX2` by using an automated process that uses Kickstart. `LINUX2` uses Fibre Channel Protocol (FCP) devices for the Red Hat Enterprise Linux installation. FCP devices are commonly known as Small Computer System Interface (SCSI) over Fibre Channel (FC). Modify the network parameters and comment the `dasda` line. You must change the lines that are marked in bold, as shown in Example 1-11 on page 24.

Example 1-11 Edit the linux2-ks.cfg Kickstart file

```
...
#version=RHEL8
# System authorization information
auth --enableshadow --passalgo=sha512

# IBM REDBOOKS RHEL8 TEMPLATE KICKSTART
# Use network installation
install
url --url="ftp://9.12.7.96/pub/rhel82"
# Use text mode install
text
#ignoredisk --only-use=dasda
# Keyboard layouts
keyboard --vckeymap=us --xlayouts='us'
# System language
lang en_US.UTF-8
# Network information
network  --bootproto=static --device=enccw0.0.0600 --gateway=9.12.4.1
--ip=9.12.7.99 --nameserver=9.12.6.6,9.12.6.7 --netmask=255.255.240.0 --noipv6
--activate  --hostname=linux2.itso.ibm.com
# Root password
rootpw --iscrypted
$6$pr46QGx7PLwzthjk$41E7GLPSsD//jHPwbQc7/CAG2SSQSkGg/pcveQUXz2IIVLOLCXH2So8n.e1
rFMjqLrfMYWifE7qY2NFfygedw/
o
...
```

1.2.7 Configuring the Virtual Network Computing server

Often, applications require a graphical environment. The Virtual Network Computing (VNC) server allows for a graphical environment to be set up easily by starting the **vncserver** service.

Complete the following steps:

1. Install the VNC server and associated packages by using the following **yum** command:

   ```
   # yum -y install tigervnc* openmotif xterm xsetroot xorg-x11-xauth
   ...
   Complete!
   ```

2. Copy the `/lib/systemd/system/vncserver\@.service` reference file to the correct location and rename it:

   ```
   # cp /lib/systemd/system/vncserver\@.service
   /etc/systemd/system/vncserver@:1.service
   ```

3. Edit **/etc/systemd/system/vncserver@:1.service** and replace the <USER> lines (in bold) in the configuration file:

   ```
   [Unit]
   Description=Remote desktop service (VNC)
   After=syslog.target network.target

   [Service]
   ```

```
Type=forking
# Clean any existing files in /tmp/.X11-unix environment
ExecStartPre=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'
ExecStart=/sbin/runuser -l root -c "/usr/bin/vncserver %i"
PIDFile=/root/.vnc/%H%i.pid
ExecStop=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'

[Install]
WantedBy=multi-user.target
```

4. Enable the VNC server at start:

```
# systemctl enable vncserver@:1.service
```

5. Set a VNC password by using the **vncpasswd** command. You need this password to connect to the VNC server:

```
# vncpasswd
Password: lnx4vm
Verify: lnx4vm
```

6. Configure the firewall:

```
# firewall-cmd --permanent --zone=public --add-service vnc-server
# firewall-cmd daemon-reload
```

7. Start the VNC server:

```
# systemctl start vncserver@:1.service
# systemctl status vncserver@:1.service

vncserver@:1.service - Remote desktop service (VNC)
   Loaded: loaded (/etc/systemd/system/vncserver@:1.service; enabled)
   Active: active (running) since Wed 2015-04-15 09:52:04 EDT; 12min ago
  Process: 1247 ExecStart=/sbin/runuser -l root -c /usr/bin/vncserver %i
(code=exited, status=0/SUCCESS)
  Process: 1233 ExecStartPre=/bin/sh -c /usr/bin/vncserver -kill %i > /dev/null
2>&1 || : (code=exited, status=0/SUCCESS)
 Main PID: 1360 (Xvnc)
   CGroup: /system.slice/system-vncserver.slice/vncserver@:1.service
           1360 /usr/bin/Xvnc :1 -desktop vmlnx2-1.itso.ibm.com:1 (root) -a...

Apr 15 09:52:11 vmlnx2-1.itso.ibm.com systemd[1]: Started Remote desktop
serv...
Hint: Some lines were ellipsized, use -l to show in full.
```

You can use the VNC client to connect to the IP address of the Linux administration system with a `:1` that is appended. A sample session is shown in Figure 1-9.

Figure 1-9 VNC client session to the VNC server

1.2.8 Copying the files that are associated with this book

To copy the files that are associated with this book to the Linux administration system, complete the following steps:

1. Change the directory to `/var/ftp/pub` if you are not already in that directory:

   # **cd /var/ftp/pub**

2. Copy the files that are associated with this book from the external FTP server that is documented on your planning worksheet by using the **curl** command. In this example, the IP address is ***9.60.87.87***.

   # **curl -O ftp://*ftpuser:linux4vm*@*9.60.87.87*/ftp/linux/24814701.tgz**

The files that are associated with this book are now copied to the Linux administration system under `/var/put/pub/SG248147/`.

1.2.9 Rebooting the system

Reboot the system to test the changes:

reboot

After your system comes back (in less than a minute), start a new SSH session to the Linux administration system.

1.3 Red Hat web console

The Red Hat Enterprise Linux web console is a Red Hat Enterprise Linux 8 web-based interface that is designed for managing and monitoring the local system, and Linux servers that are in the network environment on-premises or in the cloud.

The Red Hat Enterprise Linux web console enables various administration tasks, including the following examples:

- Managing services
- Managing user accounts
- Managing and monitoring system services
- Configuring network interfaces and firewall
- Reviewing system logs
- Managing virtual machines
- Creating diagnostic reports
- Setting kernel dump configuration
- Configuring SELinux
- Updating software
- Managing system subscriptions

The RedHat Enterprise Linux8.2 web console uses the same system APIs as are used in a terminal command window. Actions that are performed in a terminal are immediately reflected in the Red Hat Enterprise Linux web console.

You can monitor the logs of systems in the network environment and their performance, which are displayed as graphs. In addition, you can change the settings directly in the web console or through the terminal.

1.3.1 Prerequisites before accessing the web console

Before accessing the Red Hat Enterprise Linux 8.2 web console, ensure that the following prerequisites are met:

- Red Hat Enterprise Linux 8.2 is installed successfully.
- The system is registered with a suitable subscription by using the subscription manager.
- The cockpit package is installed by using the **`yum install cockpit`** command.
- The `cockpit.socket` services are enabled and started, which runs a web server, by using the **`systemctl enable --now cockpit.socket`** command.
- The firewall rules are opened or checked by using the firewalled to open the default port 9090 in the firewall by using the **`firewall-cmd --add-service=cockpit --permanent`** command and the firewalld is reloaded by using the **`firewall-cmd --reload`** command.e

1.3.2 Logging into the web console

To log into the web console, you must first access a supported web browser (for example, Chrome, Firefox, or Edge) and use the remote URL by accessing the Red Hat Enterprise Linux guest running on IBM Z (in our example, `https://129.40.23.138:9090`). Use the credentials that were used during the installation process.

In the first section of this chapter, we used `root` as a user. By using the `root` credentials, you enter them in the log-in window, as shown in Figure 1-10.

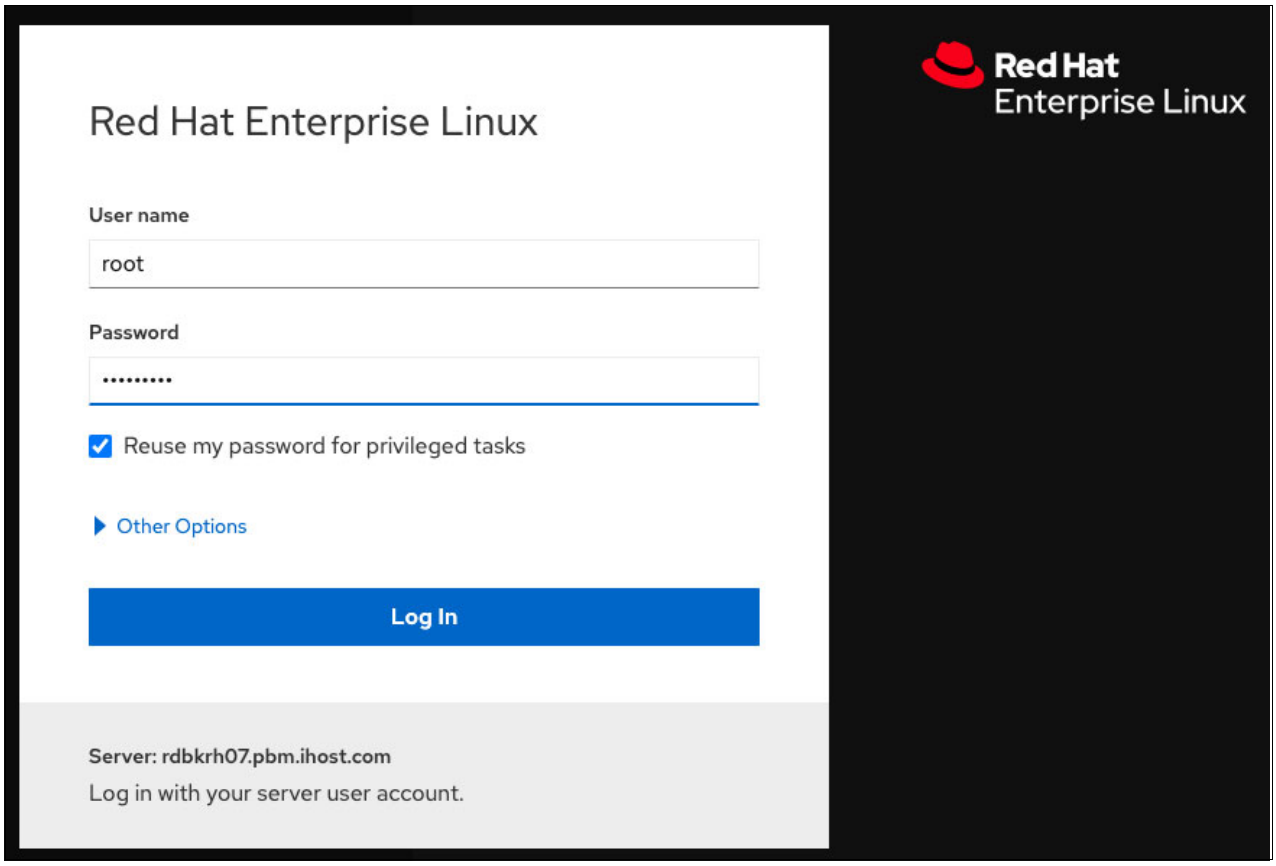

Figure 1-10 Red Hat Enterprise Linux 8.2 Web Console login

After logging into the web console, you can view the dashboard with information about the Red Hat Enterprise Linux system that is running on your environment.

To log on remotely, use the server host name and port number:

```
https://example.com:PORT_NUMBER
```

After a successful log in to the web console, the window looks similar to the example that is shown in Figure 1-11.

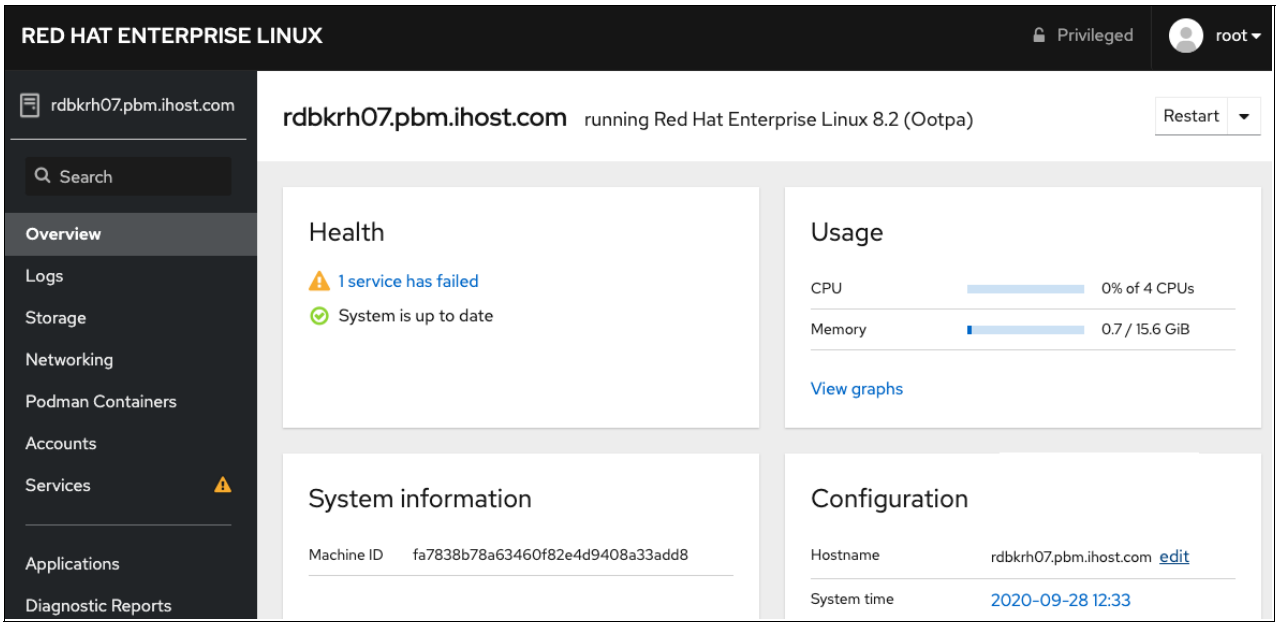

Figure 1-11 Red Hat Enterprise Linux 8.2 web console

1.3.3 Add-ons for the web console

If you need any other functions or features (see Table 1-1), you add them by using the **`yum install <add-on>`** command.

Table 1-1 Web console additional features

Feature	Package	Use
Composer	cockpit-composer	Build custom OS images
Dashboard	cockpit-dashboard	Manage multiple servers in one UI
Machines	cockpit-machines	Manage libvirt virtual machines
PackageKit	cockpit-packagekit	Software updates and application installation
podman	cockpit-podman	Manage podman containers (refer Containers chapter in the cookbook)
Session Recording	cockpit-session-recording	Recording and managing user sessions

1.3.4 System administration by using the web console

In this section, we describe how to perform basic system administration tasks, such as restart, shutdown, or basic configurations, by using the web console.

The Red Hat Enterprise Linux 8 web console is an interactive server administration interface. It interacts directly with the operating system from a real Linux session in a browser. It is helpful for a non-Linux user to perform any system level commands by using the web UI interface rather than the traditional command line.

The web console enables users to perform the following tasks:

- Monitoring basic system features, such as hardware information, time configuration, performance profiles, and connection to the realm domain.
- Inspecting system log files.
- Managing network interfaces and configuring firewall.
- Handling docker images.
- Managing virtual machines.
- Managing user accounts.
- Managing packages.
- Configuring SELinux.
- Updating software.
- Managing system subscriptions.
- Accessing the terminal.

For example, if you want to configure the host name to a more user friendly name, you assign a host name of your choice in the web console by editing the name in the overview dashboard, as shown in Figure 1-12.

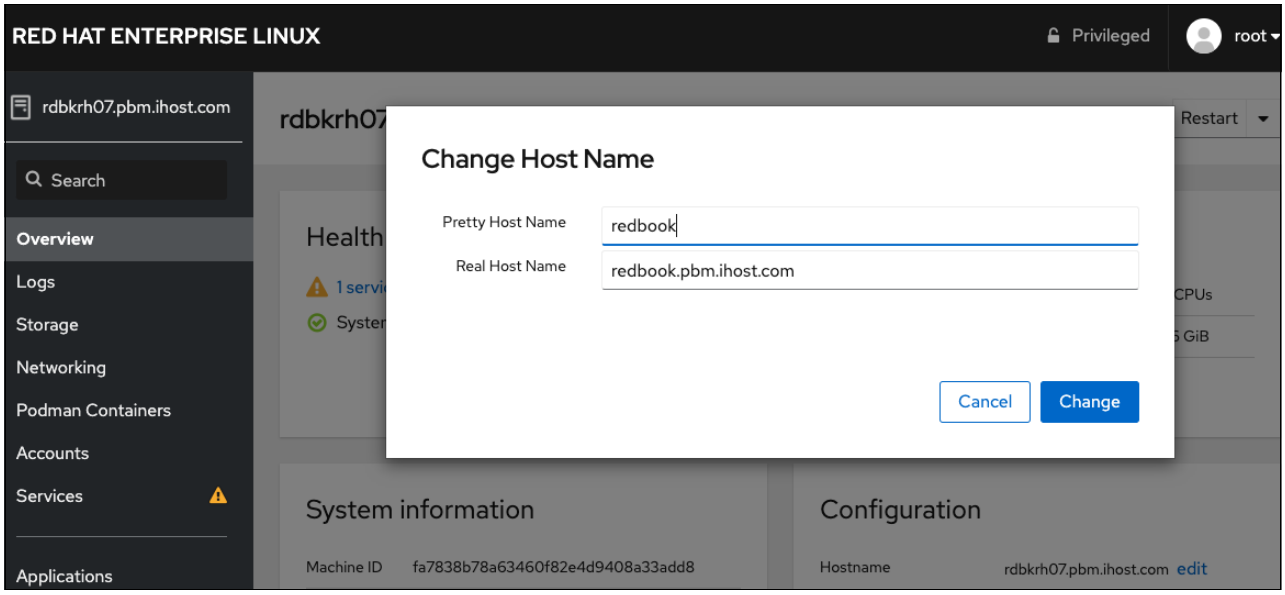

Figure 1-12 Edit host name by using the web console

1.3.5 Reviewing logs by using the web console

The Red Hat Enterprise Linux 8 web console logs section is a user interface for the `journalctl` utility. This section describes how to access system logs in the web console interface.

You can access the logs from the web console by using the `Logs` option that is in the left navigation pane, as shown in Figure 1-13.

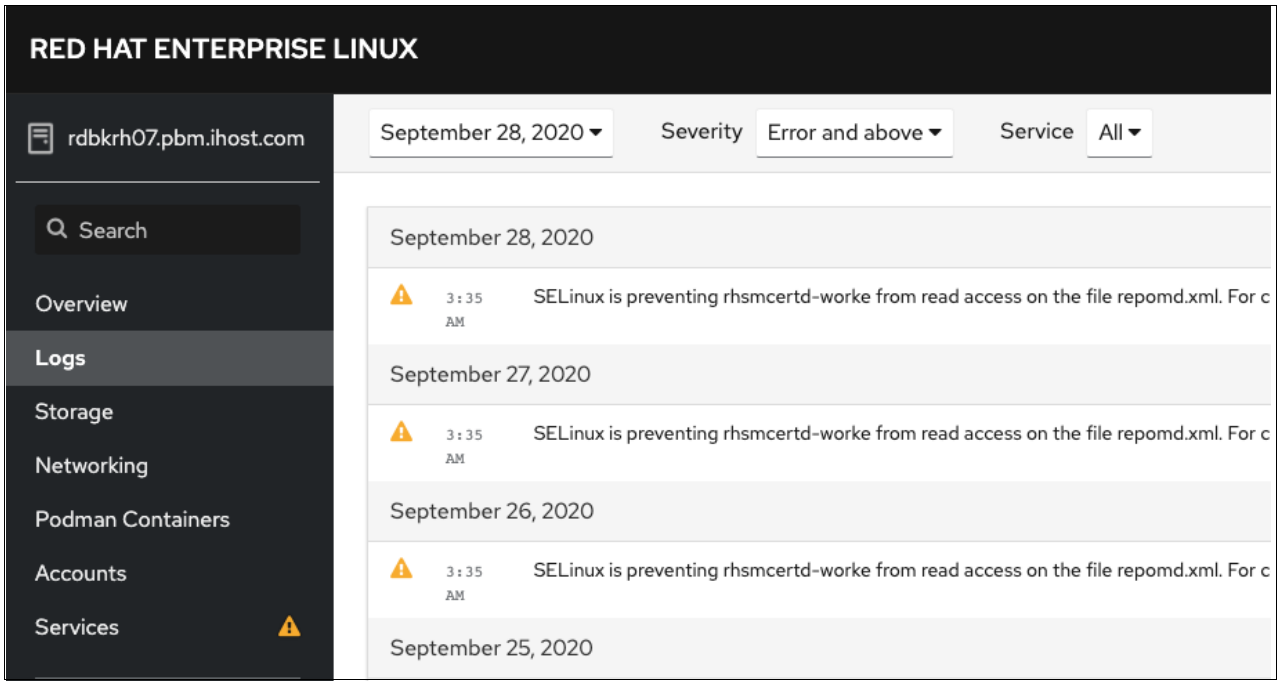

Figure 1-13 Access logs by using the web console

1.4 Application Streams in Red Hat Enterprise Linux 8.2

Application Streams (AppStream) are modules that provide multiple versions of packages with a known period of support. These modules are similar to package groups that represent an application, set of tools, or runtime languages.

Three repositories are available:

- BaseOS

  The foundation of the operating system.
- AppStream

  Supplies all of the user applications.
- Supplemental - CodeReady Linux Builder

  Developer tools and languages.

Run the **yum** command to check your environment and update your repositories, as shown in Example 1-12.

Example 1-12 yum update

```
[root@rdbkrh07 mnt]# yum update
Updating Subscription Management repositories.
RedHat Enterprise Linux 8 - BaseOS2.7 MB/s | 2.7 kB00:00
RedHat Enterprise Linux 8 - AppStream3.1 MB/s | 3.2 kB00:00
Dependencies resolved.
Nothing to do.
Complete!
[root@rdbkrh07 mnt]#
```

In this section, we describe the two main repositories: BaseOS and AppStream. BaseOS and AppStream content sets are required for a simple Red Hat Enterprise Linux installation.

1.4.1 BaseOS

Content in the BaseOS repository is intended to provide the core set of the underlying operating system function that provides the foundation for all installations. This content is available in RPM format and is subject to support terms that are similar to those terms in previous releases of Red Hat Enterprise Linux[1].

1.4.2 AppStream

Components that are made available as Application Streams can be packaged as modules or RPM packages and are delivered through the AppStream repository in Red Hat Enterprise Linux 8. Each Application Stream component features a specific lifecycle: the same as Red Hat Enterprise Linux 8 or shorter.[2]

The AppStream repository contains content packaged in two ways:

- Individual RPM packages, which are traditional known RPM packages.
- Modules, which are collections of packages that represent a logical unit: an application, language stack, database, or set of tools. These packages are built, tested, and released together. Module streams represent versions of the Application Stream components.

For more information about Application streams, see this web page.

1.4.3 Package management by using YUM in Red Hat Enterprise Linux 8.2

New in Red Hat Enterprise Linux 8 is the use of YUM for package management. This YUM is based on Dandified YUM (DNF) technology, which adds support for the new modular features.

The use of the **yum** command did not change when handling individual RPM packages. For handling the modular content, the **YUM** module command was added.

[1] https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/8.2_release_notes/distribution-of-content-in-rhel-8

[2] https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/installing_managing_and_removing_user-space_components/using-appstream_using-appstream

For more information about package management, see *Installing, managing, and removing user-space components: An introduction to AppStream and BaseOS in Red Hat Enterprise Linux 8*.

You can check the **yum** version by using the **version** command, as shown in Example 1-13.

Example 1-13 Checking the yum version

```
[root@rdbkrh07 ~]# yum install --version
4.2.17
   Installed: dnf-0:4.2.17-7.e18_2.noarch at Thu 24 Sep 2020 11:58:37 AM GMT
   Built : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla> at Thu 07 May 2020
11:34:37 AM GMT

   Installed: rpm-0:4.14.2-e18.s390x at Mon 21 Sep 2020 08:53:48 AM GMT
   Built : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla> at Fri 21 Feb 2020
01:46:16 AM GMT
[root@rdbkrh07 ~]#
```

You can display the status of a module, including enabled streams and installed profiles. In Example 1-14, we display the current status of the `postgresql-server` package.

Example 1-14 Listing modules that provide postgresql-server package

```
[root@rdbkrh07 mnt]# yum module list postgresql
Updating Subscription Management repositories.
RedHat Enterprise Linux 8 - BaseOS2.7 MB/s | 2.7 kB00:00
RedHat Enterprise Linux 8 - AppStream3.1 MB/s | 3.2 kB00:00
Last metadata expiration check: 0:00:01 ago on Tue 29 Sep 2020 08:21:32 AM EDT.
RedHat Enterprise Linux 8 - AppStream
Name        Stream   Profiles            Summary
postgresql  9.6      client, server [d]  PostgreSQL server and client module
postgresql  10 [d]   client, server [d]  PostgreSQL server and client module
postgresql  12       client, server [d]  PostgreSQL server and client module

RedHat Enterprise Linux 8 for IBM z Systems - AppStream (RPMs)
Name        Stream   Profiles            Summary
postgresql  9.6      client, server [d]  PostgreSQL server and client module
postgresql  10 [d]   client, server [d]  PostgreSQL server and client module
postgresql  12       client, server [d]  PostgreSQL server and client module

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
[root@rdbkrh07 mnt]#
```

Based on the customer requirement, you can install any specific version. For example, if you wanted to install the `postgresql` module in stream 9.6, you run the **install** command, as shown in Example 1-15.

Example 1-15 Installing postgresql

```
[root@rdbkrh07 mnt]# yum module install postgresql:9.6
Updating subscription management repositories.
RedHat Enterprise Linux 8 - BaseOS2.7 MB/s | 2.7 kB00:00
RedHat Enterprise Linux 8 - AppStream3.1 MB/s | 3.2 kB00:00
Dependencies resolved.
================================================================================
 Package Arch          Version                   Repository              Size
```

```
================================================================================
Installing group/module packages:
   postgresql-server
      s390x 9.6.10-1.module+e18+2470+d1bafa0e     InstallMedia-AppStream     4.8M
Installing dependencies:
 libpq   s390x 12.4-1.e18_2                       rhel-8-for-s390x-appstream-rpms
                                                                             189k
 postgresql
         s390x 9.6.10-1.module+e18+2470+d1bafa0e     InstallMedia-AppStream  1.4M
Installing module profiles:
 postgresql/server

Enabling module streams:
 postgresql 9.6
```

To verify the installation, run the command that is shown in Example 1-16.

Example 1-16 Verifying the desired version of postgresql is installed

```
[root@rdbkrh07 mnt]# postgres --version
   postgres (PstgreSQL) 9.6.10
[root@rdbkrh07 mnt]#
```

To remove `postgresql 9.6`, run the **`yum`** command, as shown in Example 1-17.

Example 1-17 Removing a package with the yum command

```
[root@rdbkrh07 mnt]# yum module remove postgresql:9.6
Updating Subscription Management repositories.
Last metadata expiration check: 0:21:26 ago on Tue 29 Sep 08:32:53 AM EDT.
Dependencies resolved.
================================================================================
 Package Arch           Version                    Repository               Size
================================================================================
Removing:
   postgresql server
      s390x 9.6.10-1.module+e18+2470+d1bafa0e     @InstallMedia-AppStream    22 M
Removing unused dependencies:
 libpq   s390x 12.4-1.e18_2                       rhel-8-for-s390x-appstream-rpms
                                                                             735 k
 postgresql
         s390x 9.6.10-1.module+e18+2470+d1bafa0e    @InstallMedia-AppStream 5.2 M
Disabling module profiles:
 postgresql/server

Transaction summary:
================================================================================
Remove 3 Packages

Freed space: 27 M
Is this ok [y/n]: y
Running transaction check
Transaction check succeeded
```

For more information about AppStream and BaseOS, see this web page.

2

Automated Red Hat Enterprise Linux installations by using Kickstart

In this chapter, we describe how to use Kickstart `LINUX1` and `LINUX2` from your installation server. The Linux administration system, `LNXADMIN`, is now configured as an installation server that uses FTP to share the installation tree to perform automated installations over the network. The following steps are involved in installing Linux with Kickstart for different types of Linux guests:

1. For more information about Kickstart, see this web page..
2. Configuring LINUX2 for Kickstart by using Fibre Channel Protocol devices.

Note: The Kickstart files, `linux1-ks.cfg` and `linux2-ks.cfg`, are described in 1.2.6, “Configuring Kickstart” on page 21.

This chapter includes the following topics:

2.1 Introduction

Kickstart is an automated way of installing Red Hat Enterprise Linux.

Kickstart files contain answers to all questions that normally are asked by the installation program. By using Kickstart, you can create a single file that answers all of the questions that often are asked during an interactive installation.

Kickstarting a server gives you flexibility when you are installing multiple Linux systems. Kickstart allows different package configurations and pre-installation and postinstallation scripting.

For more information about Kickstart, see this web page.

2.2 Configuring LINUX1 for Kickstart by using emulated DASD devices

Complete the following steps to configure LINUX1 for Kickstart by using emulated DASD (EDEV):

1. Log on as `LINUX1` from the 3270 console. The REXX script loads as part of the `PROFILE EXEC` and presents a message that asks to IPL disk `100`; for this step, answer `n` for no:

```
LOGON LINUX1
00: z/VM Version 7 Release 1.0, service level 2001 (64-bit),
00: built on IBM Virtualization Technology
00: There is no logmsg data
00: FILES: 0002 RDR,   NO PRT,   NO PUN
00: LOGON AT 10:02:17 EDT FRIDAY 04/17/15
00: Command complete
00: NIC 0600 is created; devices 0600-0602 defined
00: NIC 0600 is connected to VSWITCH SYSTEM VSW1
z/VM V6.3.0    2015-04-09 09:04

DMSVML2060I TCPMAINT 592 linked as 0120 file mode Z
DMSACR723I D (LNX:LNXADMIN.) R/O
DIAG swap disk defined at virtual address 300 (64988 4K pages of swap space)
DIAG swap disk defined at virtual address 301 (129980 4K pages of swap space)
Do you want to IPL Linux from minidisk 100? y/n
n
```

2. Verify that the Shared File System pools, `LNX:LINUX1` and `LNX:LNXADMIN`, are available to `LINUX1`:

```
===> q accessed
Mode  Stat     Files  Vdev  Label/Directory
A      R/W         1  DIR   LNX:LINUX1.
B      R/W         1  300   LXSWAP
C      R/W         1  301   LXSWAP
D      R/O        10  DIR   LNX:LNXADMIN.
S      R/O       698  190   MNT190
Y/S    R/O      1123  19E   MNT19E
Z      R/O       892  120   TCM592
Ready; T=0.01/0.01 17:02:35
```

3. Copy `GENERIC PRM` from the SFS disk to your local `A` disk:

```
===> copyfile generic prm d = = a
===> listfiles * * a
PROFILE  EXEC     A1
GENERIC  PRM      A1
PROFILE  XEDIT    A1
Ready; T=0.01/0.01 14:46:17
```

4. Edit the `GENERIC PRM` file (we removed the `vnc` parameter):

```
ro ramdisk_size=40000 cio_ignore=all,!condev
ip=9.12.7.98::9.12.4.1:20:vmlnx2-3.itso.ibm.com:enccw0.0.0600:none
rd.znet=qeth,0.0.0600,0.0.0601,0.0.0602,layer2=1
nameserver=9.12.6.7 nameserver=9.12.6.6
rd.dasd=0.0.0100
inst.repo=ftp://9.12.7.96/pub/rhel82

ks=ftp://9.12.7.96/pub/kickstart/linux1-ks.cfg
inst.cmdline
```

5. Run **REDHAT EXEC** to start the kickstart. Several initial kernel messages appear, which are followed by the rest of the installation process:

```
===> redhat exec
```

> **Note:** To automatically clear the 3270 console, issue the **#cp term more 0 0** command before you run **REDHAT EXEC**.

You installed Red Hat Enterprise Linux on the virtual server by using Kickstart. This process can be repeated in the future for other Linux guests.

2.3 Configuring LINUX2 for Kickstart by using Fibre Channel Protocol devices

Complete the following steps to configure LINUX2 for Kickstart by using Fibre Channel Protocol (FCP) devices:

1. Log on as `LINUX2` from the 3270 console. The REXX script loads as part of the `PROFILE EXEC` and asks to IPL disk `100`. For this step, answer `n` for no:

```
LOGON LINUX2
00: z/VM Version 7 Release 1.0, Service Level 2001 (64-bit),
00: built on IBM Virtualization Technology
00: There is no logmsg data
00: FILES: 0002 RDR,   NO PRT,   NO PUN
00: LOGON AT 10:02:17 EDT FRIDAY 04/17/15
00: Command complete
00: NIC 0600 is created; devices 0600-0602 defined
00: NIC 0600 is connected to VSWITCH SYSTEM VSW1
z/VM V6.3.0    2015-04-09 09:04

DMSVML2060I TCPMAINT 592 linked as 0120 file mode Z
DMSACR723I D (LNX:LNXADMIN.) R/O
DIAG swap disk defined at virtual address 300 (64988 4K pages of swap space)
DIAG swap disk defined at virtual address 301 (129980 4K pages of swap space)
Do you want to IPL Linux from minidisk 100? y/n
n
```

2. Verify that the Shared File System pools, `LNX:LINUX2` and `LNX:LNXADMIN`, are available to `LINUX2`:

```
===> q accessed
Mode  Stat     Files  Vdev  Label/Directory
A      R/W         3  DIR   LNX:LINUX2.
B      R/W         1  300   LXSWAP
D      R/O        11  DIR   LNX:LNXADMIN.
S      R/O       698  190   MNT190
Y/S    R/O      1123  19E   MNT19E
Z      R/O       892  120   TCM592
```

3. Copy `GENERIC PRM` from the shared disk to your local `A` disk:

```
===> copyfile generic prm d = = a
===> listfiles * * a
PROFILE  EXEC     A1
GENERIC  PRM      A1
PROFILE  XEDIT    A1
Ready; T=0.01/0.01 14:46:17
```

4. Edit the `GENERIC PRM` file (all changes are shown in bold):

```
ro ramdisk_size=40000 cio_ignore=all,!condev
ip=9.12.7.99::9.12.4.1:20:vmlnx2-4.itso.ibm.com:enccw0.0.0600:none
rd.znet=qeth,0.0.0600,0.0.0601,0.0.0602,layer2=1
nameserver=9.12.6.7 nameserver=9.12.6.6
rd.zfcp=0.0.fc00,0x500507630500c74c,0x4010401800000000
rd.zfcp=0.0.fd00,0x500507630510c74c,0x4010401800000000
inst.repo=ftp://9.12.7.96/pub/rhel71
ks=ftp://9.12.7.96/pub/kickstart/linux2-ks.cfg
inst.cmdline
```

Note: The `rd.zfcp` parameter includes the following parts:

- Virtual device; for example, `fc00`.
- Worldwide port name (WWPN) of the storage; for example, `500507630500c74c`.
- Logical unit number (LUN); for example, `4010401800000000`.

5. Run **REDHAT EXEC** to initiate the kickstart. Several initial kernel messages appear, which are followed by the rest of the installation process.

```
===> redhat exec
```

Note: To automatically clear the 3270 console, issue the **#cp term more 0 0** command before you run **REDHAT EXEC**.

6. After the installation is successful, log on to LINUX2 by using a Secure Shell (SSH) client:

```
# ssh root@9.12.7.99
root@9.12.7.99's password:
```

7. Verify the FCP configuration by using the **lsluns** command:

```
# lsluns
Scanning for LUNs on adapter 0.0.fc00
        at port 0x500507630500c74c:
                0x4010401800000000
        at port 0x50050763050bc74c:
                0x4010401800000000
Scanning for LUNs on adapter 0.0.fd00
        at port 0x500507630510c74c:
                0x4010401800000000
        at port 0x50050763051bc74c:
                0x4010401800000000
```

Although we provided only one path in the `GENERIC PRM` Conversational Monitor System (CMS) file, the FCP autoscan feature during the installation process automatically enabled all paths to the LUN that we wanted.

Note: If the output of the **lsluns** command looks like the following output, load the kernel module sg by using the **modprobe sg** command:

```
# lsluns
Scanning for LUNs on adapter 0.0.fc00
        at port 0x500507630500c74c:
                Cannot attach WLUN / LUN0 for scanning.
        at port 0x50050763050bc74c:
                Cannot attach WLUN / LUN0 for scanning.
lsluns: Error: Please load/configure SCSI Generic (sg) to use lsluns.
Scanning for LUNs on adapter 0.0.fd00
        at port 0x500507630510c74c:
                Cannot attach WLUN / LUN0 for scanning.
        at port 0x50050763051bc74c:
                Cannot attach WLUN / LUN0 for scanning.

# modprobe sg
# lsluns
Scanning for LUNs on adapter 0.0.fc00
        at port 0x500507630500c74c:
                0x4010401800000000
        at port 0x50050763050bc74c:
                0x4010401800000000
Scanning for LUNs on adapter 0.0.fd00
        at port 0x500507630510c74c:
                0x4010401800000000
        at port 0x50050763051bc74c:
                0x4010401800000000
```

Red Hat Enterprise Linux is now installed onto the virtual server by using Kickstart. You can log in to LINUX1 by using an SSH client. This process can be repeated for other Linux guests.

Note: In Step 4 on page on page 38, you modified the `GENERIC PRM` file and added the FCP devices. You can also install RHEL by using FCP devices by specifying the FCP devices only in the kickstart file instead of in the `GENERIC PRM` CMS file.

Complete the following steps:

1. Edit the `GENERIC PRM` file. In this case, you do not specify the FCP devices in this file:

```
ro ramdisk_size=40000 cio_ignore=all,!condev
ip=9.12.7.99::9.12.4.1:20:vmlnx2-4.itso.ibm.com:enccw0.0.0600:none
rd.znet=qeth,0.0.0600,0.0.0601,0.0.0602,layer2=1
nameserver=9.12.6.7 nameserver=9.12.6.6
inst.repo=ftp://9.12.7.96/pub/rhel71
ks=ftp://9.12.7.96/pub/kickstart/linux2-ks.cfg
inst.cmdline
```

2. Edit the `linux2-ks.cfg` file and add the following entries (in bold):

```
...
# Use text mode install
text
zfcp --devnum=fc00 --wwpn=500507630500c74c --fcplun=0x4010401800000000
zfcp --devnum=fd00 --wwpn=500507630510c74c --fcplun=0x4010401800000000
# Keyboard layouts
keyboard --vckeymap=us --xlayouts='us'
# System language
...
```

2.3.1 How to IPL SCSI over FCP (LINUX2)

`LINUX2` was installed directly on a Small Computer System Interface (SCSI) over FCP. To IPL a Linux guest that is installed on SCSI, complete the following steps:

1. Log on to `LINUX2` by using a 3270 terminal and answer no (`n`) to the `PROFILE EXEC` REXX script:

```
LOGON LINUX2
00: z/VM Version 7 Release 1.0, service level 2001 (64-bit),
00: built on IBM Virtualization Technology
00: There is no logmsg data
00: FILES: 0003 RDR,   NO PRT,   NO PUN
00: LOGON AT 14:40:24 EDT WEDNESDAY 04/22/15
00: Command complete
00: NIC 0600 is created; devices 0600-0602 defined
00: NIC 0600 is connected to VSWITCH SYSTEM VSW1
DMSACC724I 19E replaces Y (19E)
DMSACP723I Y (19E) R/O
z/VM V6.3.0    2015-04-09 09:04
DMSWSP100W Shared S-STAT not available
DMSWSP100W Shared Y-STAT not available
DMSVML2060I TCPMAINT 592 linked RR as 0592 file mode Z
LINUX2   AT ITSOZVM1 VIA RSCS     2015-04-22 14:40:24 EDT      WEDNESDAY
DMSACR723I D (LNX:LNXADMIN.) R/O
DIAG swap disk defined at virtual address 300 (64988 4K pages of swap space)
DIAG swap disk defined at virtual address 301 (129980 4K pages of swap space)
Do you want to IPL Linux from minidisk 100? y/n
n
```

2. Enter the following commands to IPL the virtual FCP device. The `portname` is the WWPN of the storage port for `fc00` and the LUN is the LUN definition (they are separated in blocks of eight digits):

```
===> cp set loaddev portname 50050763 0500c74c lun 40104018 00000000
===> cp ipl fc00
cp ipl fc00
00: HCPLDI2816I Acquiring the machine loader from the processor controller.
00: HCPLDI2817I Load completed from the processor controller.
00: HCPLDI2817I Now starting the machine loader.
01: HCPGSP2630I The virtual machine is placed in CP mode due to a SIGP stop and
store status from CPU 00.
00: MLOEVL012I: Machine loader up and running (version v2.4.5).
00: MLOPDM003I: Machine loader finished, moving data to final storage location.
00: Uncompressing Linux...
00: Ok, booting the kernel.
00:
...

Red Hat Enterprise Linux Server 7.1 (Maipo)
Kernel 3.10.0-229.el7.s390x on an s390x

linux2 login:
```

To automate the IPL of SCSI over FCP, use the `PROFFCP EXEC` REXX script from `LNX:LNXADMIN`. as your `PROFILE EXEC` on your local disk. Complete the following steps:

1. Log on to LINUX2 from a 3270 terminal (answer `n` to the question):

```
Logon LINUX2 using 3270 terminal:
LOGON LINUX2
00: z/VM Version 7 Release 1.0, service level 2001 (64-bit),
00: built on IBM Virtualization Technology
00: There is no logmsg data
00: FILES: 0003 RDR,   NO PRT,   NO PUN
00: LOGON AT 14:40:24 EDT WEDNESDAY 04/22/15
00: Command complete
00: NIC 0600 is created; devices 0600-0602 defined
00: NIC 0600 is connected to VSWITCH SYSTEM VSW1
DMSACC724I 19E replaces Y (19E)
DMSACP723I Y (19E) R/O
z/VM V6.3.0    2015-04-09 09:04
DMSWSP100W Shared S-STAT not available
DMSWSP100W Shared Y-STAT not available
DMSVML2060I TCPMAINT 592 linked RR as 0592 file mode Z
LINUX2   AT ITSOZVM1 VIA RSCS     2015-04-22 14:40:24 EDT      WEDNESDAY
DMSACR723I D (LNX:LNXADMIN.) R/O
DIAG swap disk defined at virtual address 300 (64988 4K pages of swap space)
DIAG swap disk defined at virtual address 301 (129980 4K pages of swap space)
Do you want to IPL Linux from minidisk 100? y/n
n
```

2. Overwrite the PROFILE EXEC on your **`LNX:LINUX2`** by using the PROFFCP EXEC file from **`LNX:LNXADMIN`**:

```
===> copyfile proffcp exec d profile exec a (replace
```

3. Edit the profile exec at **LNX:LINUX2.** Modify the REXX script variables STOWWPN, LKUPNUM, and IPLUNIT according to your Linux system:

```
===> xedit profile exec a

/* PROFILE EXEC FOR LINUX VIRTUAL SERVERS -- MOD 2015-04-09  REDBOOK1 */
/* BOOTING FROM FBA/FCP/SCSI DISKS                                    */
/**********************************************************************/
/* --- MODIFY STORAGE WWPN, LUN, IPLU VARS FOR EACH LINUX GUEST ----- */
      STOWWPN = '50050763 0500C74C'        /* 8 CHAR + SPACE + 8 CHAR */
      LKUPNUM = '40104018 00000000'        /* 8 CHAR + SPACE + 8 CHAR */
      IPLUNIT = 'FC00'                     /* IPL UNIT ADDRESS        */
/****************** NO CHANGES BELOW THIS LINE ***********************/
'CP SP CONS TO LNXADMIN START NAME 'USERID()' CONSLOG'  /* CONSLOG ON */
'CP SET RUN ON'                                  /* RUN DISCONNECTED */
'CP SET PF11 RETRIEVE FORWARD'                   /* RETRIEVE CMD FWD */
'CP SET PF12 RETRIEVE'                           /* RETRIEVE CMD BKW */
'IDENTIFY (ISODATE'                              /* IDENTIFY GUEST   */
'PIPE CP QUERY' USERID() '| VAR USER'            /* DETERMINE USERID */
 PARSE VALUE USER WITH ID . DSC .                /* CHECK IF DISCOED */

 IF ( ID <> 'LNXADMIN' ) THEN             /* IF USER IS NOT LNXADMIN */
  DO                                      /* IF USER IS NOT LNXADMIN */
   'ACCESS LNX:LNXADMIN. D'               /* IF USER IS NOT LNXADMIN */
  END                                     /* IF USER IS NOT LNXADMIN */

'SWAPGEN 0300 0524288'             /* MAKE 256M LNXSWAP VDISK AT 0300 */
'SWAPGEN 0301 1048576'             /* MAKE 512M LNXSWAP VDISK AT 0301 */
IF (DSC = 'DSC') THEN                     /* IF USER IS DISCONNECTED */
  DO
   'CP QUERY TERMINAL'
   'CP QUERY CONSOLE'
   'CP SPOOL CONSOLE STOP'                              /* CONSLOG OFF */
   'CP SET LOADDEV CLEAR PORT 'STOWWPN' LUN 'LKUPNUM   /* SET FCP VAR */
   'CP QUERY LOADDEV'                                   /* Q LOADDEV   */
   'CP IPL FC00'                                        /* BOOT LINUX  */
  END
 ELSE                                /* USER IS INTERACTIVE SO PROMPT */
  DO
   SAY 'DO YOU WANT TO IPL LINUX FROM 'IPLUNIT' AS'
   SAY 'STORAGE WWPN 'STOWWPN ' AT LOOKUP NUMBER 'LKUPNUM'? Y/N'
   PARSE UPPER PULL ANSWER .
   IF (ANSWER = 'Y') THEN
    DO
     'CP QUERY TERMINAL'
     'CP QUERY CONSOLE'
     'CP SPOOL CONSOLE STOP'                            /* CONSLOG OFF */
     'CP SET LOADDEV CLEAR PORT 'STOWWPN' LUN 'LKUPNUM /* SET FCP VAR */
     'CP QUERY LOADDEV'                                 /* Q LOADDEV   */
     'CP IPL FC00'                                      /* BOOT LINUX  */
    END
   'VMLINK TCPMAINT 592 < 592 T RR > ( NONAMES'   /* ACCESS TCP TOOLS */
  END
```

4. Verify the changes to the `PROFILE EXEC` REXX script:

```
===> ipl cms
DMSACC724I 19E replaces Y (19E)
DMSACP723I Y (19E) R/O
z/VM V6.3.0    2015-04-09 09:04

DMSWSP100W Shared S-STAT not available
DMSWSP100W Shared Y-STAT not available
LINUX2   AT ITSOZVM1 VIA RSCS     2015-04-22 18:30:12 EDT      WEDNESDAY
DMSACR723I D (LNX:LNXADMIN.) R/O
DIAG swap disk defined at virtual address 0300 (64988 4K pages of swap space)
DIAG swap disk defined at virtual address 0301 (129980 4K pages of swap space)
DO YOU WANT TO IPL LINUX FROM FC00 AS
STORAGE WWPN 50050763 0500C74C  AT LOOKUP NUMBER 40104018 00000000? Y/N
Y

00: LINEND  # , LINEDEL OFF, CHARDEL OFF, ESCAPE  " , TABCHAR OFF
00: LINESIZE 080, ATTN OFF, APL OFF, TEXT OFF, MODE VM, HILIGHT OFF
00: CONMODE 3215, BREAKIN   IMMED , BRKKEY PA1  , SCRNSAVE OFF
00: AUTOCR ON , MORE 050 010, HOLD OFF, TIMESTAMP OFF, SYS3270 OFF
00: CONS 0009 ON LDEV L0005   TERM START HOST TCPIP    FROM 9.12.5.134
00:      0009 CL T NOCONT NOHOLD COPY 001    READY FORM STANDARD
00:      0009 TO LNXADMIN RDR DIST LINUX2    FLASHC 000 DEST OFF
00:      0009 FLASH       CHAR       MDFY       0 FCB       LPP OFF
00:      0009 3215   NOEOF OPEN 0297 NOKEEP NOMSG   NAME LINUX2   CONSLOG
00:      0009 SUBCHANNEL = 0003
PORTNAME 50050763 0500C74C    LUN  40104018 00000000    BOOTPROG 0
BR_LBA   00000000 00000000
00: HCPLDI2816I Acquiring the machine loader from the processor controller.
00: HCPLDI2817I Load completed from the processor controller.
00: HCPLDI2817I Now starting the machine loader.
...

Red Hat Enterprise Linux Server 7.1 (Maipo)
Kernel 3.10.0-229.el7.s390x on an s390x

linux2 login:
```

3

Working with subscription-manager, yum, and DaNdiFied

This chapter describes how to take advantage of the Red Hat subscription model to access the suitable Red Hat Enterprise Linux channels that use the subscription-manager command line. Also described is the use of the **`yum`** or **`dnf`** command to install, remove, update, and upgrade your environment.

The Red Hat subscription offers the following advantages:

- Patches, bug fixes, updates, and upgrades
- Technical support that covers 24 x 7 availability, unlimited incidents, and multivendor case owning
- Hardware, software, and cloud provider certifications, and software assurance
- Expertise, such as security response team (SRT), customer portal, knowledge base, access labs, and training curriculum

This chapter includes the following topics:

3.1 Subscribing to a Red Hat Enterprise system by using subscription-manager

Important: Before you continue, a valid entitlement for Red Hat Enterprise Linux for IBM Z is required.

To obtain an evaluation subscription for Red Hat Enterprise Linux for IBM Z and to log on to th Red Hat customer portal, see this web page and complete the following steps:

1. Select the **Get Started** tab.
2. Select **Additional architectures**.

To download the ISO image of Red Hat Enterprise Linux 8 for IBM Z, continue with the following steps:

1. Select the Product Software tab.
2. Select **Red Hat Enterprise Linux 8**.
3. Select **Download Now**.

Red Hat introduced the Red Hat Customer Portal with the release of Red Hat Enterprise Linux 5.7, deprecating the Red Hat Network. The Red Hat Customer Portal uses a new system registration process that is called *Red Hat Subscription Manager*.

Red Hat Subscription Manager is certificate-based subscription management. With it, users can easily track subscription quantity and usage.

Important: If the system is behind an HTTP Proxy, configure it in `/etc/rhsm/rhsm.conf` as shown in the following example:

```
# an http proxy server to use (enter server FQDN)
proxy_hostname = myproxy.example.com

# port for http proxy server
proxy_port = 8080

# user name for authenticating to an http proxy, if needed
proxy_user = proxy_username

# password for basic http proxy auth, if needed
proxy_password = proxy_password
```

If your system is behind a firewall: For RHSM, allow TCP traffic over port 443 with the following Internet resources:

- For registration (RHSM, subscription-manager): `subscription.rhn.redhat.com` or `subscription.rhsm.redhat.com`
- For updates (`yum/dnf`): `cdn.redhat.com`

User can subscribe to a Red Hat Enterprise Linux system and attach a suitable Red Hat entitlement by using several methods. In this section, we describe the command-line mode because it is the most common way that customers adopt.

Complete the following steps:

1. Start a Linux terminal on the target Linux system and check the status of the system:

```
# subscription-manager status
+-------------------------------------------+
   System Status Details
+-------------------------------------------+
Overall Status: Unknown

System Purpose Status: Unknown
```

2. List the subscription and product information from this system:

```
# subscription-manager list
+-------------------------------------------+
    Installed Product Status
+-------------------------------------------+
Product Name:   Red Hat Enterprise Linux for IBM z Systems
Product ID:     72
Version:        8.2
Arch:           s390x
Status:         Unknown
Status Details:
Starts:
Ends:
```

3. Subscribe this system to a valid Red Hat account or use the credentials that were used to request the evaluation subscription for Red Hat Enterprise Linux 8 on IBM Z:

```
# subscription-manager register --username <username>
Registering to: subscription.rhsm.redhat.com:443/subscription
Password:
The system has been registered with ID: c039f5f5-8060-4ddf-8779-025ae2a0d36a
The registered system name is: rdbkrh01.pbm.ihost.com
```

4. After the system is registered, list the system's status to show that this system is registered, but it does not yet have a valid subscription that is attached to it. In our example, we used the following command:

```
# subscription-manager list
+-------------------------------------------+
    Installed Product Status
+-------------------------------------------+
Product Name:   Red Hat Enterprise Linux for IBM System z
Product ID:     72
Version: 8.2
Arch:           s390x
Status:         Not Subscribed
Status Details: Not supported by a valid subscription.
Starts:
Ends:
```

5. List all the available subscriptions. In the following example, we attach our system to a specific subscription that is identified by the pool ID 8a85f99a74d0f5f50174f3ee5e107e70:

```
# subscription-manager list --available
Subscription Name:   90 Day Supported Red Hat Enterprise Linux for IBM System z
with Smart Management Evaluation
Provides:   Red Hat Beta
```

```
                    Red Hat Developer Tools Beta (for RHEL Server for System
Z)
                    Red Hat Developer Tools (for RHEL Server for System Z)
                    Red Hat Enterprise Linux Scalable File System (for RHEL
Workstation)
                    Red Hat Ansible Engine
                    Red Hat Enterprise Linux Scalable File System (for RHEL
Server)
                    Red Hat Enterprise Linux for IBM System z (Structure A)
                    Red Hat Software Collections (for RHEL Server for System
Z)
                    Red Hat Enterprise Linux for IBM z Systems
                    Red Hat CodeReady Linux Builder for IBM z Systems
                    Red Hat Software Collections Beta (for RHEL Server for
System Z)
SKU:      RH0438617
Contract:            12473301
Pool ID:             8a85f99a74d0f5f50174f3ee5e107e70
Provides Management: Yes
Available:           200
Suggested:           1
Service Type:        L1-L3
Roles:               Red Hat Enterprise Linux Server
Service Level:       Standard
Usage:               Development/Test
Add-ons:
Subscription Type:   Instance Based
Starts:              10/04/2020
Ends:                01/01/2021
Entitlement Type:    Physical
```

Note: The **subscription-manager --list**, command might show many other subscriptions that might be available to your account. Our example shows only a cropped output that shows the 90-Day evacuation of Red Hat Enterprise Linux for IBM System z® with Smart Management.

Also, the Red Hat Enterprise Linux for IBM System z with Smart Management subscription offers many entitlements to many other channels, such as Red Hat Developer Tools, Red Hat Enterprise Linux for IBM System z, and Red Hat Ansible Engine.

6. Attach the subscription to your system by using its Pool ID number in the following command:

```
# subscription-manager attach --pool 8a85f99a74d0f5f50174f3ee5e107e70
Successfully attached a subscription for: 90 Day Supported Red Hat Enterprise
Linux for IBM System z with Smart Management Evaluation
```

7. Verify that your system is attached to a valid subscription:

```
# subscription-manager list
+-------------------------------------------+
    Installed Product Status
+-------------------------------------------+
Product Name:   Red Hat Enterprise Linux for IBM z Systems
Product ID:     72
Version:        8.2
```

```
Arch:           s390x
Status:         Subscribed
Status Details:
Starts:         10/04/2020
Ends:           01/01/2021
```

To list available repositories available to your system use the following command:

```
# subscription-manager repos --list
+----------------------------------------------------------+
    Available Repositories in /etc/yum.repos.d/redhat.repo
+----------------------------------------------------------+
Repo ID:   rhel-8-for-s390x-supplementary-source-rpms
Repo Name: Red Hat Enterprise Linux 8 for IBM z Systems - Supplementary (Source
RPMs)
Repo URL:
https://cdn.redhat.com/content/dist/rhel8/$releasever/s390x/supplementary/sourc
e/SRPMS
Enabled:   0

Repo ID:   ansible-2.9-for-rhel-8-s390x-rpms
Repo Name: Red Hat Ansible Engine 2.9 for RHEL 8 s390x (RPMs)
Repo URL:
https://cdn.redhat.com/content/dist/layered/rhel8/s390x/ansible/2.9/os
Enabled:   0
....
```

To enable more repositories for your system, use the following command:

```
# subscription-manager repos --enable=ansible-2.9-for-rhel-8-s390x-rpms
Repository 'ansible-2.9-for-rhel-8-s390x-rpms' is enabled for this system.
```

For more information about the Subscription Manager command-line options, see this web page.

For more information about Subscription Manager, see this Red Hat documentation web page.

3.2 Using the yum package manager

The **yum** command is the traditional tool for getting, installing, deleting, querying, and otherwise managing Red Hat Enterprise Linux RPM software packages from official Red Hat software repositories, and other third-party repositories.

Note: Consider the following points:

- Our examples use a direct connected system to the Red Hat Customer Portal. For production systems, Red Hat recommends the use of the Red Hat Satellite, which is included with the Smart Management suspicions for Red Hat Enterprise Linux. For more information about a complete lifecycle management solution, see this web page.
- Your system might require the Red Hat GNU Privacy Guard (GPG) key to install packages from the Red Hat repositories. To have the suitable key installed, use the following command:

  ```
  # rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
  ```

Red Hat Enterprise Linux 8 includes the yum tool version 4 (YUMv4), which is based on DNF technology. The **yum** command can be used to install, upgrade, and update packages in your system.

By default, `yum` always installs the latest version. If a specific version is needed, you might be required to download and install it manually. You also can specify multiple package names (which are separated by spaces) on the command line. The **yum install** command installs the package if it is not present. If a package includes any dependencies, `yum` automatically installs them for you.

For software installation, the **yum** command and most of its options work the same way in Red Hat Enterprise Linux 8 as they did in Red Hat Enterprise Linux 7.

The older Python API that is provided by YUM v3 is no longer available. You can migrate your plug-ins and scripts to the new API that is provided by YUM v4 (DNF Python API), which is stable and fully supported by Red Hat.

For more information, see this web page.

Note: The **rpm** command also can be used to query installed packages, as shown in the following example:

```
# rpm -qa | grep <package_name>
```

The configuration file for `yum` can be found in `/etc/yum.conf`.

The `/etc/yum.conf` configuration file contains one mandatory `[main]` section, where YUM options have a global effect, as shown in Example 3-1.

Example 3-1 Showing the content of the /etc/yum.conf file

```
[main]
gpgcheck=1
installonly_limit=3
clean_requirements_on_remove=True
best=True
skip_if_unavailable=False
```

Note: The configuration file `/etc/yum.conf` is a symbolic link to `/etc/dnf/dnf.conf`, which is similar to the following example:

```
lrwxrwxrwx. 1 root root 12 Feb 18  2020 /etc/yum.conf -> dnf/dnf.conf
```

As a best practice, define individual repositories in new or existing `.repo` files in the `/etc/yum.repos.d/` directory.

As an example of extra repositories that can be added to yum, in addition to official Red Hat repositories, we show a popular repository that is called Extra Packages for Enterprise Linux (EPEL). This repository is a volunteer-based community effort from the Fedora project to create a repository of high-quality add-on packages that complement the Red Hat Enterprise Linux server and its compatible spin-offs, such as CentOS.

Example 3-2 shows the contents of the `epel.repo` file after you add the repository information so that yum can use this newly added external repository.

Important: Although all the extra packages from EPEL does not conflict with Red Hat Enterprise Linux, the packages are *not* supported by Red Hat. The extra packages from EPEL can be used for experimental purposes and must not be used in any production systems.

For more information about how to install EPEL, see this web page. For more information about their FAQ, see this web page.

Example 3-2 Sample /etc/yum.repos.d/epel.repo file

```
[epel]
name=Extra Packages for Enterprise Linux $releasever - $basearch
#baseurl=https://download.fedoraproject.org/pub/epel/$releasever/Everything/$basea
rch
metalink=https://mirrors.fedoraproject.org/metalink?repo=epel-$releasever&arch=$ba
search&infra=$infra&content=$contentdir
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-8
```

Note: The values that you define in the `[main]` section of the `/etc/yum.conf` file might override the values that are set in individual `[repository]` sections.

The following examples show how to use yum:

- Use the following command to install a package:

```
# yum install nano
Updating Subscription Management repositories.
Last metadata expiration check: 0:08:09 ago on Sun 04 Oct 2020 11:51:12 AM EDT.
Dependencies resolved.
================================================================================
 Package                 Architecture              Version
Repository                                         Size
================================================================================
Installing:
 nano                                s390x
2.9.8-1.el8                         InstallMedia-BaseOS
581 k
Transaction Summary
================================================================================
Install  1 Package
Total size: 581 k
Installed size: 2.3 M
Is this ok [y/N]:
```

- Use the following command to upgrade installed packages:

```
# yum upgrade s390utils
Updating Subscription Management repositories.
Last metadata expiration check: 0:00:21 ago on Sun 04 Oct 2020 11:51:12 AM EDT.
Package s390utils-2:2.6.0-28.el8.s390x is already installed.
Dependencies resolved.
================================================================================
 Package                    Architecture         Version
Repository                                          Size
```

```
================================================================================
Upgrading:
 s390utils                                s390x
2:2.6.0-28.el8_2.2               rhel-8-for-s390x-appstream-rpms
49 k
 s390utils-base                           s390x
2:2.6.0-28.el8_2.2                rhel-8-for-s390x-baseos-rpms
1.2 M
 s390utils-cmsfs                          s390x
2:2.6.0-28.el8_2.2               rhel-8-for-s390x-appstream-rpms
71 k
 s390utils-cpuplugd                       s390x
2:2.6.0-28.el8_2.2               rhel-8-for-s390x-appstream-rpms
71 k
 s390utils-iucvterm                       s390x
2:2.6.0-28.el8_2.2               rhel-8-for-s390x-appstream-rpms
92 k
 s390utils-mon_statd                      s390x
2:2.6.0-28.el8_2.2               rhel-8-for-s390x-appstream-rpms
61 k
 s390utils-osasnmpd                       s390x
2:2.6.0-28.el8_2.2               rhel-8-for-s390x-appstream-rpms
60 k
 s390utils-ziomon                         s390x
2:2.6.0-28.el8_2.2                rhel-8-for-s390x-appstream-rpms
155 k

Transaction Summary
================================================================================
Upgrade  8 Packages

Total download size: 1.8 M
Is this ok [y/N]:
y
```

- Use the following command to update your whole system:

```
# yum update
Updating Subscription Management repositories.
Last metadata expiration check: 0:25:41 ago on Sun 04 Oct 2020 01:23:45 PM EDT.
Dependencies resolved.
================================================================================
 Package                      Architecture   Version
Repository                                   Size
================================================================================
Installing:
 kernel                                           s390x
4.18.0-193.19.1.el8_2
rhel-8-for-s390x-baseos-rpms              2.8 M
 kernel-core                                      s390x
4.18.0-193.19.1.el8_2
rhel-8-for-s390x-baseos-rpms               19 M
 kernel-modules                                   s390x
4.18.0-193.19.1.el8_2
rhel-8-for-s390x-baseos-rpms              3.7 M
Upgrading:
```

```
 NetworkManager
...
unbound-libs                                      s390x
1.7.3-11.el8_2
rhel-8-for-s390x-appstream-rpms           462 k
 yum                                               noarch
4.2.17-7.el8_2
rhel-8-for-s390x-baseos-rpms              193 k
 zlib                                              s390x
1.2.11-16.el8_2
rhel-8-for-s390x-baseos-rpms              105 k
Installing dependencies:
 libslirp                                          s390x
4.3.0-3.module+el8.2.1+6816+bedf4f91
rhel-8-for-s390x-appstream-rpms            63 k

Transaction Summary
================================================================================
Install    4 Packages
Upgrade  101 Packages

Total download size: 262 M
Is this ok [y/N]:
```

- Use the following command to query information about installed or not installed packages:

```
# yum info dnf
Updating Subscription Management repositories.
Last metadata expiration check: 0:28:09 ago on Sun 04 Oct 2020 01:23:45 PM EDT.
Installed Packages
Name         : dnf
Version      : 4.2.17
Release      : 6.el8
Architecture : noarch
Size         : 1.6 M
Source       : dnf-4.2.17-6.el8.src.rpm
Repository   : @System
From repo    : anaconda
Summary      : Package manager
URL          : https://github.com/rpm-software-management/dnf
License      : GPLv2+ and GPLv2 and GPL
Description  : Utility that allows users to manage packages on their systems.
             : It supports RPMs, modules and comps groups & environments.
```

- Use the following command to search packages:

```
yum search <<package>>
```

What if we want to install the library to access the ICA hardware crypto on IBM Z? You can use yum to search for a generic term (in this case, “crypto”) and verify whether a match exists:

```
# yum search crypto
Updating Subscription Management repositories.
Last metadata expiration check: 0:47:16 ago on Sun 04 Oct 2020 01:23:45 PM EDT.
================================================================== Name &
Summary Matched: crypto
==================================================================
```

```
erlang-crypto.s390x : Cryptographical support
cryptopp-doc.noarch : Documentation for cryptopp
crypto-policies.noarch : System-wide crypto policies
...
libica.s390x : Library for accessing ICA hardware crypto on IBM z Systems
..
```

The yum search function searches the name of the packages and in the description of the package.

Note: You can search for a specific binary, such as c library; as shown in the following example:

```
# yum provides */stdin.h
```

This search returns all packages that provide the `stdin.h` library.

For more information about yum commands, see this web page.

3.3 Introduction to DaNdiFied yum

DaNdiFied (DNF) is a package manager for RPM-based Linux distributions that installs, updates, and removes packages. It was first introduced in Fedora 18 and today is becoming the next-generation version of the traditional yum package manager. It has more advanced and robust features than yum. The following features distinguish DNF from yum:

- Dependency calculation that is based on modern dependency-solving technology
- Optimized memory-intensive operations
- The ability to run in Python 2 and Python 3
- Complete documentation available for Python APIs

YUM v4, which is included with Red Hat Enterprise Linux, uses the DNF technology behind the scenes. However, you also can use DNF commands directly, if needed.

For more information about DNF, see this web page.

DNF features a similar syntax to yum, as shown in the following examples:

- Use the following command to install packages:

```
# dnf install firefox
Updating Subscription Management repositories.
Last metadata expiration check: 0:03:51 ago on Sun 04 Oct 2020 02:36:40 PM EDT.
Dependencies resolved.
================================================================================
 Package                        Architecture          Version
Repository                                          Size
================================================================================
Installing:
 firefox                                    s390x
78.3.0-1.el8_2                  rhel-8-for-s390x-appstream-rpms
97 M
Upgrading:
```

```
 nspr                                          s390x
4.25.0-2.el8_2                  rhel-8-for-s390x-appstream-rpms
139 k
 nss                                           s390x
3.53.1-11.el8_2                 rhel-8-for-s390x-appstream-rpms
670 k
 nss-softokn                                   s390x
3.53.1-11.el8_2                 rhel-8-for-s390x-appstream-rpms
463 k
 nss-softokn-freebl                            s390x
3.53.1-11.el8_2                 rhel-8-for-s390x-appstream-rpms
260 k
 nss-sysinit                                   s390x
3.53.1-11.el8_2                rhel-8-for-s390x-appstream-rpms
71 k
 nss-util                                      s390x
3.53.1-11.el8_2                 rhel-8-for-s390x-appstream-rpms
133 k
Installing dependencies:
 libICE                                        s390x
1.0.9-15.el8                    InstallMedia-AppStream
71 k
 libSM                                         s390x
1.2.3-1.el8                     InstallMedia-AppStream
47 k
 libXt                                         s390x
1.1.5-12.el8                    InstallMedia-AppStream
174 k
 liberation-fonts-common                       noarch
1:2.00.3-7.el8                  InstallMedia-BaseOS
25 k
 liberation-sans-fonts                         noarch
1:2.00.3-7.el8                  InstallMedia-BaseOS
610 k
 mozilla-filesystem                            s390x
1.9-18.el8                      InstallMedia-AppStream
11 k
 redhat-indexhtml                noarch              8-7.el8
InstallMedia-BaseOS                                242 k

Transaction Summary
================================================================================
Install  8 Packages
Upgrade  6 Packages

Total size: 100 M
Total download size: 98 M
Is this ok [y/N]:
```

- Use the following command to install package groups:

```
# dnf groupinstall "Security Tools"
Updating Subscription Management repositories.
Last metadata expiration check: 0:02:04 ago on Sun 04 Oct 2020 02:36:40 PM EDT.
Dependencies resolved.
```

```
================================================================================
 Package                                    Architecture
Version                               Repository
Size
================================================================================
Installing group/module packages:
 scap-security-guide                        noarch
0.1.48-7.el8                          InstallMedia-AppStream
6.9 M
Installing dependencies:
 GConf2                                     s390x
3.2.6-22.el8                          InstallMedia-AppStream
1.0 M
 libxslt                                    s390x
1.1.32-4.el8                          InstallMedia-BaseOS
237 k
 openscap                                   s390x
1.3.2-6.el8                           InstallMedia-AppStream
3.3 M
 openscap-scanner                           s390x
1.3.2-6.el8                           InstallMedia-AppStream
68 k
 xml-common                                 noarch
0.6.3-50.el8                          InstallMedia-BaseOS
39 k
Installing Groups:
 Security Tools

Transaction Summary
================================================================================
Install  6 Packages

Total size: 12 M
Installed size: 247 M
Is this ok [y/N]:
```

- Use the following command to remove packages:

```
# dnf remove httpd
Updating Subscription Management repositories.
Dependencies resolved.
================================================================================
 Package                  Architecture        Version
Repository                                    Size
================================================================================
Removing:
 httpd                                s390x
2.4.37-21.module+el8.2.0+5008+cca404a3
@InstallMedia-AppStream                     4.3 M
Removing unused dependencies:
 apr                     s390x             1.6.3-9.el8
@InstallMedia-AppStream                     305 k
 apr-util                 s390x             1.6.1-6.el8
@InstallMedia-AppStream                     243 k
```

```
 apr-util-bdb              s390x           1.6.1-6.el8
@InstallMedia-AppStream                    11 k
 apr-util-openssl           s390x           1.6.1-6.el8
@InstallMedia-AppStream                    20 k
 httpd-filesystem                      noarch
2.4.37-21.module+el8.2.0+5008+cca404a3
@InstallMedia-AppStream                   400
 httpd-tools                           s390x
2.4.37-21.module+el8.2.0+5008+cca404a3
@InstallMedia-AppStream                   204 k
 mod_http2                             s390x
1.11.3-3.module+el8.2.0+4377+dc421495
@InstallMedia-AppStream                   500 k
 redhat-logos-httpd          noarch          81.1-1.el8
@InstallMedia-BaseOS                      3.3 k

Transaction Summary
================================================================================
Remove  9 Packages

Freed space: 5.6 M
Is this ok [y/N]:
```

For more use case examples, API references and more information about DNF, see this web page.

Note: By default, Red Hat Enterprise Linux 8.x uses YUM v4, which is based on DNF technologies. The use of DNF commands directly is an option, but we highly recommend the use of YUM v4.

4

Networking

This chapter introduces the system administrator to the tasks that are required to support networking on a Red Hat Enterprise Linux on IBM Z. It provides an overview of the processes that are required to configure virtual network interface Controller (vNICs) and perform the required tailoring steps to support guest operating system virtual machines.

It also describes the steps to manage network devices and define a minimal TCP/IP configuration.

This chapter includes the following topics:

4.1 Basic concepts

The following basic concepts are used throughout this chapter:

- Central processor complex (CPC): A physical collection of hardware that consists of main storage, one or more central processors, timers, and channels.
- Linux guest: A Linux server that is running under z/VM that can run applications separate from other guests.
- LPAR: A logical partition that is created at the firmware or microcode level of a System z processor. Typically, z/VM runs in the LPAR; then, Linux guests run under z/VM, but it is possible to run Linux directly in an LPAR.
- Hypervisor: A system that allows multiple operating systems to share a single hardware. For z/VM, it creates a layer to manage the dispatching of virtual guests.
- Channel command word (CCW): The original I/O operation that is used for communications with the channel subsystem. The CCW contains a channel command, such as read, write, or control, along with the data address of the data area involved. The data is passed to the channel subsystem.

  The channel subsystem communicates status of the I/O back to the issuing application. When a channel communicates with an application in an asynchronous fashion, it is referred to as a *channel interrupt*.
- Open System Adapter (OSA): A hardware network controller that you can install in a mainframe I/O cage. The adapter integrates several hardware features and supports many networking transport protocols. The OSA card is the strategic communications device for the mainframe architecture. It includes several key features that distinguish it from CCW-based communications.
- OSA QDIO Queued Direct I/O: A highly efficient data transfer architecture, which dramatically improves data transfer speed and efficiency for transmission control protocol/internet protocol (TCP/IP) traffic.
- vmcp: The virtual machine control programmer (vmcp) command allows Linux users to substitute **`vmcp`** for the line end character plus **`cp`** to issue CP commands from an SSH or virtual console of the Linux guest.
- IUCV: The Inter-User Communications Vehicle (IUCV) is a z/VM CP interface for passing data between virtual machines or between the CP interface and a virtual machine.
- Channel Path Identifier (CHPID): The channel, represented by a channel path ID or CHPID, represents the communication path. A CHPID is the handle by which communication between the CPC and an external device is facilitated.

  CHPID number is associated with a physical channel port location (PCHID), and a logical channel subsystem. The CHPID number range is still 00 - FF and must be unique within a logical channel subsystem. These resources can be dedicated or shared among LPARs. They are defined in the I/O configuration data set (IOCDS).

4.2 Connectivity options

This section provides an overview of connectivity options that are available on the IBM Z platform.

The z/VM hypervisor uses virtualization to allow system administrators to manage resources on the IBM Z platform. This virtualization layer provides flexibility, availability, and security capabilities for Linux guests while creating an isolated and protected environment for all types of applications, including OpenShift, Ansible, and containers.

The virtual network that is provided by z/VM for Linux guest communication provides high throughputs and better reliability (failure tolerance).

Typically, the Z platform provides the following networking options:

- OSA-Express features (1 GbE, 10 GbE, 25 GbE, and 1000BaseT). They can be directly attached to Red Hat Enterprise Linux or with a zVM Virtual switch (vSWITCH).
- RoCE Express: RoCE is RDMA over Converged Ethernet. These cards can operate in three different protocol modes (TCP/IP, RDMA, or SMC-R). They are available in 10 GbE and 25 GbE options.

  zVM Virtual switch is *not* an option with these cards, but they can be virtualized to the guest with PCI Passthru. SMC-R operating mode features restrictions, such as no routing, and only TCP protocol. SMC-R can be used by a ZVM hypervisor and a Linux guest.
- HiperSockets: A virtual hardware device for high-speed low latency transfers with large MTU sizes possible. These HiperSockets are restricted to “in-the-box” communications between partitions.
- Internal Shared Memory (ISM): This virtual hardware device can be used directly by the Linux Guest. ISM is designed to use shared memory areas to provide low-latency, high-bandwidth, cross-LPAR connections for applications.

These options provide Linux on IBM Z guests the ability to communicate over the network. In general, these Linux servers use virtual devices as their own physical network.

Linux servers typically use virtual switches to access the network. A virtual switch allows grouping of several OSA-Express devices to create one logical link for providing fault-tolerance and high-speed connections between the physical OSA devices and the Linux guests.

In general, decisions regarding the best methods for networking are based on reliability, performance, and availability. You have multiple alternatives to couple your Red Hat Enterprise Linux guests to the network. In this chapter, we provide information about how to set up Red Hat Enterprise Linux servers.

Red Hat Enterprise Linux 8.2 includes the following network device drivers to provide connections:

- qeth (OSA-Express and HiperSockets in QDIO mode)
- LCS (alternative driver):
- CTCM
- AF_IUCV
- SMC
- RoCE
- ISM
- HiperSockets

Hint: The **znetconf** command can be used to automatically define qeth, LCS, and CTCM devices.

The qeth driver

The qeth driver is the most common network driver on IBM Z.

Communications between OSA-Express devices and the qeth device driver are available by using the Queued Direct I/O (QDIO) protocol. These network devices are represented by a folder under the `/sys` file system when the qeth module is loaded.

Note: The qeth files (`/sys/bus/ccwgroup/drivers/qeth`) are created when the qeth module loads.

The qeth device driver supports a multitude of network connections, for example, connections through Open Systems Adapters (OSA), HiperSockets, guest LANs, and virtual switches.

The qeth device driver requires three I/O subchannels (Channel Command Word [CCW] devices for read, write, and data) for each OSA-Express CHPID that is predefined in your IOCDS. This book does not cover the configuration of these devices in IOCDS.

The qeth device driver also requires three I/O subchannels for each HiperSocket CHPID or OSA-Express CHPID in QDIO mode. One subchannel is for control reads, one for control writes, and the third is for data (see Table 4-1).

Table 4-1 Three I/O subchannels definition

Name	Description	Example
read_device_id	Must be even	0.0.0640
write_device_id	Must be the device bus-ID of the read subchannel plus one	0.0.0641
data_device_id	Might be the device bus-ID of the write subchannel plus one	0.0.0642

The qeth device driver uses the QDIO protocol to communicate with the HiperSockets and OSA-Express adapter (see Figure 4-1).

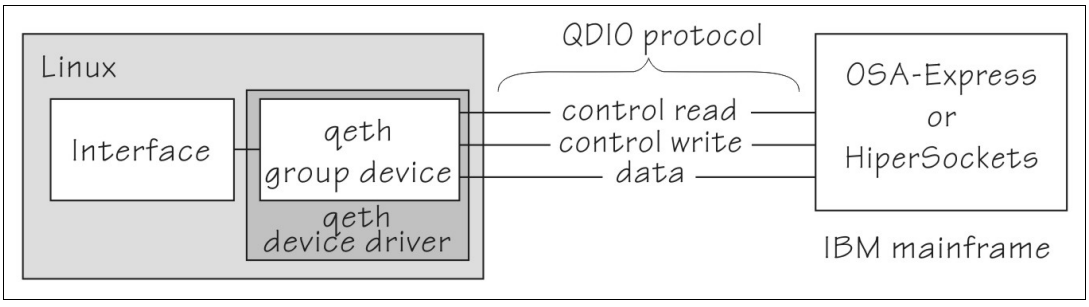

Figure 4-1 qeth representation

You can manually define the qeth network device by using the following commands:

```
cio_ignore -r 0.0.0640,0.0.0641,0.0.0642
echo 0.0.0640,0.0.0641,0.0.0642 > /sys/bus/ccwgroup/drivers/qeth/group
```

You can also use the following **znetconf** command:

```
cio_ignore -r 0.0.0640,0.0.0641,0.0.0642
```

```
znetconf -a 0.0.f503
```

Also, you can use the **chzdev** command:

```
chzdev -e qeth 0.0.0640,0.0.0641,0.0.0642 layer2=1
```

For more information about znetconf, see this web page.

LAN channel station driver

The LAN channel station (LCS) device driver supports 1000Base-T Ethernet on the OSA-Express2 and OSA-Express 3 features. It is another option to couple Red Hat Enterprise Linux servers in the network.

No module parameters are available for the LCS device driver.

Before you set up LCS devices, ensure that you load the LCS module by using the **modprobe lcs** command.

To define an LCS group device, write the device bus-IDs of the subchannel pair to `/sys/bus/ccwgroup/drivers/lcs/group`. by running the following command:

```
# echo <read_device_bus_id>,<write_device_bus_id> >
/sys/bus/ccwgroup/drivers/lcs/group
```

As an example, assume that 0.0.0740 is the device bus-ID that corresponds to a read subchannel, you issue the following command:

```
# echo 0.0.0740,0.0.0740 > /sys/bus/ccwgroup/drivers/lcs/group
```

Use the **znetconf** command to set up the LCS devices. If needed, the **znetconf** command attempts to load the device driver.

CTCM driver

The CTCM driver supports channel-to-channel (CTC) connections to allow a high-speed point-to-point communication between two IBM Z system instances. Although CTC devices are popular in z/OS and z/VM, these connections are deprecated for Linux setups. Therefore, avoid using them.

AF_IUCV

The address family (AF) z/VM Inter-User Communications Vehicle (IUCV) is similar to CTC in that it is a point-to-point connection. It is used extensively when z/VM components must communicate with each other. IUCV support was added into the Linux kernel to allow connections between Linux servers. IUCV was the other option for Linux network connectivity.

IUCV facilitates connections between applications across Linux virtual machines within a z/VM system. In particular, an application that runs on Linux on z/VM can communicate with:

- Itself
- Other applications that run on the same Linux instance
- Applications that are running on:
  - Other instances of Linux on z/VM, within the same z/VM system
  - A z/VM guest other than Linux, within the same z/VM system
- The z/VM control program (CP)

If you are familiar with configuring an ICIC (IBM Cloud® Infrastructure Center) or CMA (Cloud Manager Appliance) on a Z cloud environment, you can be comfortable configuring IUCV for your Red Hat Enterprise Linux guests.

For more information about how to set up IUCV, see this web page.

Shared Memory Communication

In addition to the OSA-Express features, IBM Z offers Remote Direct Memory Access over Converged Ethernet (RoCE) features. RoCE supports the Shared Memory Communication-Remote (SMC-R) protocol, which allows operating systems to communicate through shared memory across platforms. SMC-R offers high performance, low latency network options. As with OSA-Express, the RoCE-Express features provide 25 GbE and 10 GbE options.

SMC is the latest IBM Z network feature for faster communication between application workloads within the same CPC footprint. With Red Hat Enterprise Linux 8.2, you can take advantage of Shared Memory Communications Direct (SMC-D) to improve performance and boost network connections. This feature is beneficial when Linux runs colocated with IBM z/OS® on IBM Z.

SMC connections can be defined without requiring physical resources, such as RoCE adapters, PCI bandwidth, ports, I/O slots, network resources, and 10 GbE switches.

The SMC protocol can be used for connections through:

- SMCs through RDMA (SMC-R) with RoCE devices.
- SMCs Direct (SMC-D) with ISM devices

If both variants are available for a connection, SMC-D is used.

SMC-R and SMC-D use shared memory to provide low-latency, high-bandwidth, cross-LPAR connections for applications. This support is intended to provide application-transparent direct memory access (DMA) communications to TCP endpoints for socket-based connections.

RoCE

Linux servers on IBM Z can take advantage of SMCs over the Remote Direct Memory Access (RDMA) function to optimize network connectivity for mission critical workloads by using SMC technology. A network that uses RDMA provides better performance than an adapter that is used as a NIC for network-intensive applications.

RoCE provides for a high-speed connection interface. Adapters that support RoCE must be installed in both nodes.

The ports on the RoCE adapters can be cabled directly together or connected through a switch.

Red Hat Enterprise Linux servers use the technological advances and high-performing nature of the I/O processing that is offered by RoCE with IBM Z servers.

ISM

Internal Shared Memory (ISM) is a virtual PCI network adapter that enables direct access to shared virtual memory, which provides a highly optimized network interconnect for Z platform intra-communications. SMC-D uses ISM.

HiperSocket

HiperSocket is another memory-to-memory communication option, which is available between LPARs within the IBM Z platform. HiperSockets is an integrated function of the firmware that uses an internal QDIO (iQDIO) architecture to provide an efficient and secure internal network. Because it is an internal network, HiperSockets avoid the cost of the physical network infrastructure

4.3 Network considerations

OSA and HiperSockets are supported by multiple operating systems, such as z/OS, z/VM, and Linux on IBM Z.

OSA and RoCE can be used to send traffic between CECs.

HiperSockets and SMC-D support only traffic between LPARs on a single CEC. SMC-R and HiperSockets provide similar performance benefits. SMC-D/ISM provides advantages beyond both.

4.4 Working with an attached OSA card

In this section, we describe how to connect a Red Hat Enterprise Linux server to the network by way of QDIO CHPID E4. For complex environments, assigning OSA addresses to multiple servers can become a complex task; therefore, a VSWITCH must be used to connect Linux servers to the network.

First, in the IOCDS hardware definition, you must define CHPID E4 as an OSA device with type QDIO. You also must give the z/VM LPAR ARIES32 (RDBKZVM2) access to devices 1E40 - 1E4F of this channel, as shown in the following steps to connect a Linux server to an Ethernet network.

From a z/VM x3270 terminal, complete the following steps:

1. Verify that the path to the OSA devices is in an ONLINE status by using the command that is shown in Example 4-1.

Example 4-1 Verifying the path

```
QUERY CHPID E4
06:45:39 Path E4 online to devices 1E40 1E41 1E42 1E43 1E44 1E45 1E46 1E47
06:45:39 Path E4 online to devices 1E48 1E49 1E4A 1E4B 1E4C 1E4D 1E4E 1E4F
Ready; T=0.01/0.01 21:01:13
```

2. Display the OSA devices to confirm that they are available to the z/VM system guests (see Example 4-2).

Example 4-2 Displaying the OSA devices

```
QUERY 1E40-1E4F
06:47:28 OSA  1E40 ATTACHED TO DTCVSW1  0600 DEVTYPE OSA         CHPID E4 OSD
06:47:28 OSA  1E41 ATTACHED TO DTCVSW1  0601 DEVTYPE OSA         CHPID E4 OSD
06:47:28 OSA  1E42 ATTACHED TO DTCVSW1  0602 DEVTYPE OSA         CHPID E4 OSD
06:47:28 OSA  1E43 FREE    , OSA  1E44 FREE    , OSA  1E45 FREE    , OSA  1E46
FREE
```

```
06:47:28 OSA  1E47 FREE    , OSA  1E48 FREE    , OSA  1E49 FREE    , OSA  1E4A
FREE
06:47:28 OSA  1E4B FREE    , OSA  1E4C FREE    , OSA  1E4D FREE    , OSA  1E4E
FREE
06:47:28 OSA  1E4F FREE
Ready; T=0.01/0.01 06:47:28
```

The output that is shown in Example 4-2 on page 65 shows the used and available OSA addresses. We assign addresses 1E43, 1E44, and 1E45 addresses to a Red Hat Enterprise Linux server.

3. To attach these addresses, use the following CP commands:

Note: In this example, we assign virtual address 1E00, 1E01, and 1E02 to 1E43, 1E44, and 1E45 to standardize the device addresses in our Linux instances. Inside Linux, these virtual addresses are used instead of the real address numbers.

```
QUERY 1E43-1E45
06:57:37 OSA  1E43 FREE    , OSA  1E44 FREE    , OSA  1E45 FREE
Ready; T=0.01/0.01 06:57:37

ATTACH 1E43 RDBKRH05 1E00
06:57:41 OSA 1E43 ATTACHED TO RDBKRH05 1E00
Ready; T=0.01/0.01 06:57:41

ATTACH 1E44 RDBKRH05 1E01
06:57:50 OSA 1E44 ATTACHED TO RDBKRH05 1E01
Ready; T=0.01/0.01 06:57:50

ATTACH 1E45 RDBKRH05 1E02
06:57:59 OSA 1E45 ATTACHED TO RDBKRH05 1E02
Ready; T=0.01/0.01 06:57:59

QUERY 1E43-1E45
06:58:05 OSA  1E43 ATTACHED TO RDBKRH05 1E00 DEVTYPE OSA         CHPID E4 OSD
06:58:05 OSA  1E44 ATTACHED TO RDBKRH05 1E01 DEVTYPE OSA         CHPID E4 OSD
06:58:05 OSA  1E45 ATTACHED TO RDBKRH05 1E02 DEVTYPE OSA         CHPID E4 OSD
Ready; T=0.01/0.01 06:58:05
```

4. Make these settings permanent in the user directory by using the following commands:

```
DIRM FOR RDBKRH05 DEDICATE 1E00 1E43
DIRM FOR RDBKRH05 DEDICATE 1E01 1E44
DIRM FOR RDBKRH05 DEDICATE 1E02 1E45
```

To connect to your Linux server by using SSH, configure the network interfaces to your subnet (in our case, 192.168.0.0/24) in Red Hat Enterprise Linux by using the following commands:

```
# cio_ignore -r 1e00-1e02

# lscss |grep 1731
0.0.0640 0.0.0006  1732/01 1731/01 yes  80  80  ff   02000000 00000000
0.0.0641 0.0.0007  1732/01 1731/01 yes  80  80  ff   02000000 00000000
0.0.0642 0.0.0008  1732/01 1731/01 yes  80  80  ff   02000000 00000000
0.0.1e00 0.0.0018  1732/01 1731/01      80  80  ff   e8000000 00000000
0.0.1e01 0.0.0019  1732/01 1731/01      80  80  ff   e8000000 00000000
0.0.1e02 0.0.001a  1732/01 1731/01      80  80  ff   e8000000 00000000
```

```
# znetconf -a 1e00
Scanning for network devices...
Successfully configured device 0.0.1e00 (enc1e00)

# lsqeth enc1e00
Device name                     : enc1e00
------------------------------------------------------------------------
        card_type               : OSD_10GIG
        cdev0                   : 0.0.1e00
        cdev1                   : 0.0.1e01
        cdev2                   : 0.0.1e02
        chpid                   : E4
        online                  : 1
        portname                : no portname required
        portno                  : 0
        state                   : UP (LAN ONLINE)
        priority_queueing       : always queue 2
        buffer_count            : 64
        layer2                  : 1
        isolation               : none
        bridge_role             : none
        bridge_state            : inactive
        bridge_hostnotify       : 0
        bridge_reflect_promisc  : none
        switch_attrs            : unknown
        vnicc/flooding          : 0
        vnicc/learning          : 0
        vnicc/learning_timeout  : 600
        vnicc/mcast_flooding    : 0
        vnicc/rx_bcast          : 1
        vnicc/takeover_learning : 0
        vnicc/takeover_setvmac  : 0

# nmcli connection add type ethernet con-name enc1e00 ifname enc1e00
connection.autoconnect yes
Connection 'enc1e00' (e5cf73ad-da5d-41c4-ae66-33cb22f78597) successfully added

# nmcli connection modify enc1e00 ipv4.method manual ipv4.addresses
192.168.0.136/24 ipv4.dns '129.40.106.1,129.40.106.2' ipv4.dns-search
pbm.ihost.com

# nmcli con up enc1e00
nmcli con up enc1e00
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/14)
```

4.5 Working with a dual network interface card

When you configure your Linux server to communicate with other networks over two different vNICs, you might need to add some custom routes settings because the router of the second IP address is not set in the virtual machine automatically.

You rarely need to have multiple network interfaces on a system to reach all the IP addresses on the system by using external subnets. However, If you plan to connect your Linux server to two networks through two different virtual NICs, you must set policy-based routing to ensure that two different gateways can be used at the same time.

The Linux server supports a dual NIC configuration, which allows you to configure one NIC for internal use and use the other NIC for more public exchanges.

If multiple NICs in Linux are configured with different addresses, communication into that subnet by default always is done by the first NIC in that subnet that is started. The problem occurs because in Linux, only one default gateway is used. Therefore, even if the network packet can reach the second NIC, the response packet still uses the default gateway. At that point, the response packet cannot reach the sender.

To obtain an expected behavior, split routing must be set up. Complete the following steps to set up a dual NIC:

1. Enable a second NIC (0620).

   Note: We added the following two lines into our user directory:

   ```
   NICDEF 0620 TYPE QDIO LAN SYSTEM VSWITCH2
   NICDEF 0620 VLAN 401
   ```

   a. If you do not want to log off and log on to the Linux server to make the device available, issue the following commands to manually set it up:

      On VM as `MAINT`:

      ```
      SET VSWITCH VSWITCH2 GRANT RDBKRH04 VLAN 401
      ```

      On Linux:

      ```
      # vmcp define nic 620 type qdio dev 3
      NIC 0620 is created; devices 0620-0622 defined

      # vmcp couple 620 system vswitch2
      NIC 0620 is connected to VSWITCH SYSTEM VSWITCH2
      ```

   b. Make the new network device available in Linux by using the following commands:

      ```
      # cio_ignore -r 620-622

      # lscss
      Device   Subchan.  DevType CU Type Use  PIM PAM POM  CHPIDs
      ----------------------------------------------------------------------
      0.0.7000 0.0.0000  1732/05 1731/05 yes  80  80  ff   f0000000 00000000
      0.0.7001 0.0.0001  1732/05 1731/05 yes  80  80  ff   f0000000 00000000
      0.0.7002 0.0.0002  1732/05 1731/05 yes  80  80  ff   f0000000 00000000
      0.0.0620 0.0.0006  1732/01 1731/01      80  80  ff   01000000 00000000
      0.0.0621 0.0.0007  1732/01 1731/01      80  80  ff   01000000 00000000
      0.0.0622 0.0.0008  1732/01 1731/01      80  80  ff   01000000 00000000
      ```

```
0.0.0640 0.0.0009  1732/01 1731/01 yes  80  80  ff   02000000 00000000
0.0.0641 0.0.000a  1732/01 1731/01 yes  80  80  ff   02000000 00000000
0.0.0642 0.0.000b  1732/01 1731/01 yes  80  80  ff   02000000 00000000
0.0.0100 0.0.000d  3390/0e 3990/e9 yes  80  80  ff   ff000000 00000000
0.0.0102 0.0.000e  3390/0e 3990/e9 yes  80  80  ff   ff000000 00000000
0.0.0103 0.0.000f  3390/0e 3990/e9 yes  80  80  ff   ff000000 00000000
0.0.0104 0.0.0010  3390/0e 3990/e9 yes  80  80  ff   ff000000 00000000
0.0.0105 0.0.0011  3390/0e 3990/e9 yes  80  80  ff   ff000000 00000000
0.0.0009 0.0.0012  0000/00 3215/00 yes  80  80  ff   ff000000 00000000
0.0.0300 0.0.001d  9336/10 6310/80 yes  80  80  ff   ff000000 00000000
0.0.0301 0.0.001e  9336/10 6310/80 yes  80  80  ff   ff000000 00000000

# chzdev -e qeth 0.0.0620,0.0.0621,0.0.0622 layer2=1
QETH device 0.0.0620:0.0.0621:0.0.0622 configured

# lszdev qeth
TYPE  ID                          ON   PERS  NAMES
qeth  0.0.0620:0.0.0621:0.0.0622  yes  yes   enc620
qeth  0.0.0640:0.0.0641:0.0.0642  yes  no    enc640

# nmcli dev
DEVICE   TYPE      STATE         CONNECTION
enc640   ethernet  connected     System enc640
enc7000  ethernet  connected     enc7000
enc620   ethernet  disconnected  --
lo       loopback  unmanaged     --
```

2. Set an IP address for the new network device by using the following commands:

```
# nmcli connection add type ethernet con-name enc620 connection.autoconnect yes
Connection 'enc620' (d4973c7e-8d59-4230-8169-5ec746fce163) successfully added.

# nmcli connection modify enc620 ipv4.method manual ipv4.addresses
10.12.6.135/24

# nmcli con up enc620
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/8)
```

3. Determine which is the default gateway and which NIC needs another route table added. Run the following command:

```
# ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enc640: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 02:01:02:00:00:0c brd ff:ff:ff:ff:ff:ff
    inet 129.40.23.135/24 brd 129.40.23.255 scope global noprefixroute enc640
       valid_lft forever preferred_lft forever
    inet6 fe80::1:2ff:fe00:c/64 scope link
       valid_lft forever preferred_lft forever
```

```
3: enc7000: <BROADCAST,MULTICAST,NOARP,UP,LOWER_UP> mtu 8192 qdisc mq state UP
group default qlen 1000
    link/ether 0e:00:f0:32:00:02 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.7/24 brd 192.168.1.255 scope global noprefixroute enc7000
       valid_lft forever preferred_lft forever
    inet6 fe80::70d9:a232:6fd6:9c63/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
4: enc620: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 02:01:02:00:00:0b brd ff:ff:ff:ff:ff:ff
    inet 10.12.6.135/24 brd 10.12.6.255 scope global noprefixroute enc620
       valid_lft forever preferred_lft forever
    inet6 fe80::2bbd:5025:6f9e:8ff5/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

4. At this point, the virtual machine now has two NICs: `enc640` with an IP address of `129.40.23.135`, and `enc620` with an IP address of `10.12.6.135`. Use the following command to check the route table:

```
# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         129.40.23.254   0.0.0.0         UG    100    0        0 enc640
10.12.6.0       0.0.0.0         255.255.255.0   U     102    0        0 enc620
129.40.23.0     0.0.0.0         255.255.255.0   U     100    0        0 enc640
192.168.1.0     0.0.0.0         255.255.255.0   U     101    0        0 enc7000
```

   Therefore, the NIC (`enc640`) has a default gateway that can be reached from the outside; `enc620` does not have a gateway, so it cannot be reached from other networks.

5. You must add another route table for enc620. Use the following command to assign a unique routing table number (table ID) to the connection profile by using the `ipv4.route-table` property:

```
# nmcli con mod enc620 ipv4.route-table 10
```

6. Set up the IP routing policy rules by configuring the connection's `ipv4.routing-rules` property by using the following commands:

```
# nmcli con mod enc620 ipv4.routing-rules "priority 100 iif enc620 table 10"
# nmcli con mod enc620 +ipv4.routing-rules "priority 110 from 10.12.6.135 table
10"
# nmcli con mod enc620 +ipv4.routing-rules "priority 120 to 10.12.6.0/24 table
10"
```

7. After modifying all of the connection properties, put the settings into place by using the following **`nmcli dev reapply`** command:

```
# nmcli dev reapply enc620
Connection successfully reapplied to device 'enc620'.
```

8. List the IP rules table by using the following commands:

```
# ip rule show
0: from all lookup local
100:from all iif enc620 lookup 10
110:from 10.12.6.135 lookup 10
32766:from all lookup main
32767:from all lookup default

# ip route show table 10
10.12.6.0/24 dev enc620 proto kernel scope link src 10.12.6.135 metric 102
```

```
# ip route show table all | grep "table" | sed 's/.*\(table.*\)/\1/g' | awk
'{print $2}' | sort | uniq
10
local
```

9. To test, use the following commands:

```
# ping -I enc620 10.12.6.68
PING 10.12.6.68 (10.12.6.68) from 10.12.6.135 enc620: 56(84) bytes of data.
64 bytes from 10.12.6.68: icmp_seq=1 ttl=59 time=0.963 ms
64 bytes from 10.12.6.68: icmp_seq=2 ttl=60 time=0.093 ms
64 bytes from 10.12.6.68: icmp_seq=3 ttl=59 time=0.209 ms
64 bytes from 10.12.6.68: icmp_seq=4 ttl=60 time=0.091 ms
```

4.6 Working with virtual switches

A virtual switch (vSWITCH) is a software program that enables one virtual host to communicate with another virtual host within a computer system. vSWITCHs typically emulate functions of a physical Ethernet switch.

In Linux on IBM Z, a VSWITCH provides direct attachment of z/VM guests to the local physical network segment. The VSWITCH allows IP network architects and network administrators to treat z/VM guests as a server in the network.

Most z/VM virtualization involves abstracting and sharing real hardware among multiple guests. vSWITCHs do not physically exist. vSWITCHs use real OSA hardware to transmit data to and from the “outside” world, but that architecture is not apparent to the guests connecting to the vSWITCH.

The VSWITCH method allows IBM Z guests to connect over the network. This method is efficient and secure. It eliminates the need to have a z/VM service machine or a Linux guest acting as a router, which reduces the effect on z/VM to perform this role.

The zVM system administrators are responsible for defining the VSWITCH. In our environment, they defined it by using the following command:

```
DEFINE VSWITCH VSWITCH3 ETHERNET RDEV 1E84.P0 1EE4.P0 ,
  VLAN 400 PORTNAME OSA1E80 OSA1EE0
```

The system administrators then add this command into the `SYSTEM.CONFIG` file to make this setting survive any VM IPLs (reboots).

Complete the following steps from a zVM x3270 terminal to couple a Red Hat Enterprise Linux server to the network by using VSWITCH:

1. Get information about VSWITCH3:

```
QUERY VSWITCH VSWITCH3
07:37:37 VSWITCH SYSTEM VSWITCH3   Type: QDIO    Connected: 3    Maxconn:
INFINITE
07:37:37   PERSISTENT  RESTRICTED    ETHERNET                  Accounting: OFF
07:37:37   USERBASED LOCAL
07:37:37   VLAN Aware  Default VLAN: 0400    Default Porttype: Access  GVRP:
Enabled
07:37:37               Native  VLAN: 0001    VLAN Counters: OFF
07:37:37   MAC address: 02-01-02-00-00-03    MAC Protection: Unspecified
```

```
07:37:37   IPTimeout: 5         QueueStorage: 8
07:37:37   Isolation Status: OFF        VEPA Status: OFF
07:37:37  Uplink Port:
07:37:37   State: Ready                          PriQueuing: OFF
07:37:37   PMTUD setting: EXTERNAL   PMTUD value: 9000     Trace Pages: 8
07:37:37   Portname: OSA1E80  RDEV: 1E84.P00 Controller: DTCVSW4  VDEV: 0606
ACTIVE
07:37:37         Adapter ID: 85610002B7F8.01AC
07:37:37         EQID: ETHSWIT3
07:37:37  Portname: OSA1EE0  RDEV: 1EE4.P00 Controller: NONE    VDEV: NONE
Error: Portname conflict
07:37:37         Adapter ID: 85610002B7F8.0214
07:37:37         EQID: ETHSWIT3
Ready; T=0.01/0.01 07:37:37
```

2. Add the following NICDEF statements into each server's user directory to allow them to access VLAN 400 by way of VSWITCH3.

```
NICDEF 0640 TYPE QDIO LAN SYSTEM VSWITCH3
NICDEF 0640 VLAN 400
```

To define and bring a network device online in your Red Hat Enterprise Linux server, you can follow the instructions in section "Defining network configuration on Red Hat Enterprise Linux servers" on page 89. Remember to replace the device number and IP address to those used in your network environment.

4.7 Working with SMC interfaces

SMC-D and SMC-R use shared memory to provide low-latency, high-bandwidth, and cross-LPAR connections for applications. This support is intended to provide application-transparent direct memory access (DMA) communications to TCP endpoints for socket-based connections.

Linux guests with a dedicated PCI function are not eligible for Live Guest Relocation. In addition, if in an SSI environment, a guest can use PCIe functions only if they are in a singleton domain or a domain where the suitable PCIe facilities hardware is available on all members in the domain. Also, PCI must be enabled by using the **FEATURES ENABLE PCI** statement in the system configuration file on all members in the domain.

For more information, see *z/VM V7.1 CP Planning and Administration*, SC24-6271.

Our RoCE environment is shown in Table 4-2.

Table 4-2 SMC-R Environment

Server name	LPAR	RoCE address	OSA CHPID	OSA Real Channels	(enc1e00) IP address
rdbkrh05	RDBKZVM2	000000D2	E2	1E24-1E26	10.12.6.136
rdbkrh13	RDBKZVM1	000000D1	E2	1E24-1E26	10.12.6.149

Note: SMC traffic requires a mapping of the OSA or HiperSockets network interfaces to the RoCE adapters or ISM devices.

zVM Linux guests require PNET IDs to be assigned to OSA, HiperSocket, RoCE, and ISM devices through the IOCDS. Example 4-3 shows a sample of our IOCP definition.

Example 4-3 IOCP definition.

```
# OSA
CHPID PATH=(CSS(0,1,2,3),E2),SHARED,                            *
               PARTITION=((CSS(0),(ARIES01),(=)),(CSS(1),(ARIES18,ARIES*
               19,ARIES1A,ARIES1B),(=))),                         *
               NOTPART=((CSS(2),(ARIES2C,ARIES2D,ARIES2E,ARIES2F),(=)))*
               ,PCHID=130,PNETID=PERFNET,TYPE=OSD
   CNTLUNIT CUNUMBR=1E20,                                        *
               PATH=((CSS(0),E2),(CSS(1),E2),(CSS(2),E2),(CSS(3),E2)), *
               UNIT=OSA
         IODEVICE ADDRESS=(1E20,015),UNITADD=00,CUNUMBR=(1E20),UNIT=OSA
         IODEVICE ADDRESS=(1E2F,001),UNITADD=FE,CUNUMBR=(1E20),       *
               UNIT=OSAD

# ISM and RoCE
FUNCTION FID=C1,VF=17,PCHID=158,PNETID=PERFNET,                 *
               PART=((ARIES31),(=)),TYPE=ROC2,PORT=1
          FUNCTION FID=C2,VF=18,PCHID=158,PNETID=PERFNET,           *
               PART=((ARIES32),(=)),TYPE=ROC2,PORT=1
          FUNCTION FID=D1,VF=17,PCHID=158,PNETID=PERFNET,           *
               PART=((ARIES31),(=)),TYPE=ROC2,PORT=2
          FUNCTION FID=D2,VF=18,PCHID=158,PNETID=PERFNET,           *
               PART=((ARIES32),(=)),TYPE=ROC2,PORT=2
          FUNCTION FID=E1,VF=17,PCHID=1B8,PNETID=PERFNET,           *
               PART=((ARIES31),(=)),TYPE=ROC2,PORT=1
          FUNCTION FID=E2,VF=18,PCHID=1B8,PNETID=PERFNET,           *
               PART=((ARIES32),(=)),TYPE=ROC2,PORT=1
          FUNCTION FID=F1,VF=17,PCHID=1B8,PNETID=PERFNET,           *
               PART=((ARIES31),(=)),TYPE=ROC2,PORT=2
          FUNCTION FID=F2,VF=18,PCHID=1B8,PNETID=PERFNET,           *
               PART=((ARIES32),(=)),TYPE=ROC2,PORT=2
          FUNCTION FID=1030,VF=39,VCHID=7C0,PNETID=PERFNET,         *
               PART=((ARIES31),(=)),TYPE=ISM
          FUNCTION FID=1031,VF=40,VCHID=7C0,PNETID=PERFNET,         *
               PART=((ARIES32),(=)),TYPE=ISM
```

In Linux, the matching PNETID associates the ISM device with an Ethernet device.

Changes are required in two places in the SYSTEM CONFIG file on your z/VM LPARs to enable PCI functions:

- In the FEATURES ENABLE section, add PCI
- Add the STORAGE section immediately after the FEATURES section

Example 4-4 shows how the changes look in our SYSTEM CONFIG file. We highlighted the settings in bold.

Example 4-4 Sample SYSTEM CONFIG file

```
/*********************************************************************/
/*                        Features Statement                         */
/*********************************************************************/
```

```
 Features ,
   Retrieve ,                        /* Retrieve options              */
     Default  20 ,                   /* Default.... default is 20     */
     Maximum  255 ,                  /* Maximum.... default is 255    */
   MaxUsers noLimit ,                /* No limit on number of users   */
   Vdisk Userlim 2097350 blocks,     /* Maximum vdisk allowed per user */
         Syslim Infinite,
   Disconnect_Timeout 15,            /* Can be OFF, default is 15 min  */

   Enable ,                          /* Enable the following features  */
     New_Devices_Initialized_When_Added,  /* Make new devices online */
     STP_Timezone,                   /* STP feature is used           */
     PCI,

   Disable ,                         /* Disable the following features */
     Set_Privclass ,                 /* Disallow SET PRIVCLASS command */
     Auto_Warm_IPL ,                 /* Prompt at IPL always          */
     Clear_TDisk   ,                 /* Don't clear TDisks at IPL time */
     Validate_Shutdown               /* Don't require system name     */


/*********************************************************************/
/*                        Storage  Statement                         */
/*********************************************************************/

 Storage ,
   IOAT  64 Megabytes Warn 80 Percent
```

After a z/VM IPL, you must vary devices online by using the following command:

```
VARY ON PCIF <device_number>
```

Attaching PCI devices

Complete the following steps to attach the PCI devices for RoCE and ISM:

1. On zVM RDBKZVM2 LPAR, verify the expected PCI functions are available by using the following **QUERY** PCIFUNCTION (**PCIF**) command:

   ```
   QUERY PCIF
   18:41:06 PCIF 000000C2 FREE                        DISABLED RoCE Express 2
   18:41:06 PCIF 000000D2 FREE                        DISABLED RoCE Express 2
   18:41:06 PCIF 000000E2 FREE                        DISABLED RoCE Express 2
   18:41:06 PCIF 000000F2 FREE                        DISABLED RoCE Express 2
   18:41:06 PCIF 00001031 FREE                        DISABLED ISM
   Ready; T=0.01/0.01 18:41:06
   ```

   The output indicates that the PCI devices are in DISABLED state.

2. Verify the IOAT sub-pool settings by using the following **QUERY FRAMES** command:

   ```
   QUERY FRAMES
   18:43:23 All Frames:
   18:43:23    Configured=16777216  Real=16777216  Usable=16777216  Offline=0
   18:43:23    Pageable=16569773  NotInitialized=0  GlobalClearedAvail=8242
   18:43:23    LocalClearedAvail=296  LocalUnclearedAvail=236
   18:43:23    LockedPageable=0% LockedWarning=--% LockedFail=--%
   18:43:23    IOATSize=16384 IOATUsed=0% IOATMaxused=0% IOATWarning=80%
   ```

```
18:43:23
18:43:23 Frames < 2G:
18:43:23    GlobalUnclearedAvail=500879  Pageable=517019
LogicalFreeStorage=2134
18:43:23    RealFreeStorage=4  LockedRS=39  LockedCmd=0
18:43:23    MinidiskCache=3  Nucleus/Prefix=3599  Trace=1225  Other=267
18:43:23
18:43:23 Frames > 2G:
18:43:23    GlobalUnclearedAvail=15640342  Pageable=16052754
LogicalFreeStorage=2958
18:43:23    RealFreeStorage=73  LockedRS=806  LockedCmd=0
18:43:23    MinidiskCache=5486  Other=196337
Ready; T=0.01/0.01 18:43:23
```

Note: You can verify the storage locking and warning settings in the output of the **QUERY FRAMES** command.

3. Take your server down (in our case, rdbkrh05) by using the following command:

   ```
   CP SIGNAL SHUTDOWN RDBKRH05 WITHIN 60
   ```

4. Add the following ATTACH statement to the zVM user directory to attach the 002D PCI device:

   ```
   COMMAND ATTACH PCIF 00D2 TO &USERID
   ```

5. Check the status of PCI devices *before* bringing the server up by using the following command:

   ```
   Q PCIF
   11:49:06 PCIF 000000C2 FREE                          DISABLED RoCE Express 2
   11:49:06 PCIF 000000D2 FREE                          DISABLED RoCE Express 2
   11:49:06 PCIF 000000E2 FREE                          DISABLED RoCE Express 2
   11:49:06 PCIF 000000F2 FREE                          DISABLED RoCE Express 2
   11:49:06 PCIF 00001031 FREE                          DISABLED ISM
   Ready; T=0.01/0.01 11:49:06
   ```

6. Bring your Linux server up (in our case, rdbkrh05) by using the following command:

   ```
   XAUTOLOG RDBKRH05
   ```

7. Confirm the status of RoCE device:

   ```
   QUERY PCIF
   12:37:51 PCIF 000000C2 FREE                          DISABLED RoCE Express 2
   12:37:51 PCIF 000000D2 ATTACHED TO RDBKRH05 000000D2 ENABLED  RoCE Express 2
   12:37:51 PCIF 000000E2 FREE                          DISABLED RoCE Express 2
   12:37:51 PCIF 000000F2 FREE                          DISABLED RoCE Express 2
   12:37:51 PCIF 00001031 FREE                          DISABLED ISM
   Ready; T=0.01/0.01 12:37:51
   ```

Repeat these steps on your next server (in our case, rdbkrh13 on the RDBKZVM1 LPAR). For more information about the RoCE address number and server details, see Table 4-2 on page 72.

Installing the SMC tools package

To support SMC-D (ISM) and SMC-R (RoCE), you must install the `smc-tools` package in all servers. Use the command that is shown in the following example to install this package:

```
# yum install -y smc-tools
... Output Ommited ....
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
 Preparing     :
1/1
 Installing    : smc-tools-1.2.2-3.el8.s390x
1/1
 Running scriptlet: smc-tools-1.2.2-3.el8.s390x
1/1
 Verifying     : smc-tools-1.2.2-3.el8.s390x
1/1
Installed products updated.

Installed:
 smc-tools-1.2.2-3.el8.s390x

Complete!
```

Install SMC tools on `rdbkrh05` and `rdbkrh13` servers.

Defining the OSA card

In our environment, E2 and E4 chpids were defined to include the PNET ID. In our environment, the value is PERFNET. We decided to use E2 for the network device.

On VM

Complete the following steps to define a new NIC on Red Hat servers:

1. Query CHPID E2 to get device numbers:

```
QUERY CHPID E2
18:06:56 Path E2 online to devices 1E20 1E21 1E22 1E23 1E24 1E25 1E26 1E27
18:06:56 Path E2 online to devices 1E28 1E29 1E2A 1E2B 1E2C 1E2D 1E2E 1E2F
Ready; T=0.01/0.01 18:06:56
```

2. Confirm if the devices are available:

```
QUERY 1E20 1E21 1E22 1E23 1E24 1E25 1E26 1E27 1E28 1E29 1E2A 1E2B 1E2C 1E2D
1E2E 1E2F
18:13:01 OSA  1E20 ATTACHED TO DTCVSW4  0600 DEVTYPE OSA         CHPID E2 OSD
18:13:01 OSA  1E21 ATTACHED TO DTCVSW4  0601 DEVTYPE OSA         CHPID E2 OSD
18:13:01 OSA  1E22 ATTACHED TO DTCVSW4  0602 DEVTYPE OSA         CHPID E2 OSD
18:13:01 OSA  1E23 FREE    , OSA  1E24 FREE    , OSA  1E25 FREE    , OSA  1E26
FREE
18:13:01 OSA  1E27 FREE    , OSA  1E28 FREE    , OSA  1E29 FREE    , OSA  1E2A
FREE
18:13:01 OSA  1E2B FREE    , OSA  1E2C FREE    , OSA  1E2D FREE    , OSA  1E2E
FREE
18:13:01 OSA  1E2F FREE
```

```
Ready; T=0.01/0.01 18:13:01
```

Note: The first address must be an even number; therefore, we use 1E88, 1E89, and 1E8A (real devices).

3. Update Linux server user directory by issuing:

```
DIRM FOR RDBKRH05 DEDICATE 1E00 1E24
DIRM FOR RDBKRH05 DEDICATE 1E01 1E25
DIRM FOR RDBKRH05 DEDICATE 1E02 1E26
```

Note: We define a 1E00 virtual network interface (VNICs).

4. Dynamically attach the devices by using the following commands:

```
ATTACH 1E24 RDBKRH05 1E00
18:16:44 OSA 1E24 ATTACHED TO RDBKRH05 1E00
Ready; T=0.01/0.01 18:16:44

ATTACH 1E25 RDBKRH05 1E01
18:16:51 OSA 1E25 ATTACHED TO RDBKRH05 1E01
Ready; T=0.01/0.01 18:16:51

ATTACH 1E26 RDBKRH05 1E02
18:16:57 OSA 1E26 ATTACHED TO RDBKRH05 1E02
Ready; T=0.01/0.01 18:16:57
```

On Linux server

Complete the following steps:

1. Enable 1e00 device:

```
# cio_ignore -r 1e00-1e02

# chzdev -e qeth 0.0.1e00,0.0.1e01,0.0.1e02 layer2=1 buffer_count=128
QETH device 0.0.1e00:0.0.1e01:0.0.1e02 configured

# lsqeth enc1e00
Device name                     : enc1e00
------------------------------------------------------------------------------
        card_type               : OSD_10GIG
        cdev0                   : 0.0.1e00
        cdev1                   : 0.0.1e01
        cdev2                   : 0.0.1e02
        chpid                   : E2
        online                  : 1
        portname                : no portname required
        portno                  : 0
        state                   : UP (LAN ONLINE)
        priority_queueing       : always queue 2
        buffer_count            : 128
        layer2                  : 1
        isolation               : none
        bridge_role             : none
        bridge_state            : inactive
        bridge_hostnotify       : 0
        bridge_reflect_promisc  : none
```

```
        switch_attrs             : unknown
        vnicc/flooding           : 0
        vnicc/learning           : 0
        vnicc/learning_timeout   : 600
        vnicc/mcast_flooding     : 0
        vnicc/rx_bcast           : 1
        vnicc/takeover_learning  : 0
        vnicc/takeover_setvmac   : 0
```

2. Configure the new network interface (1e00):

```
# nmcli connection add type ethernet con-name enc1e00 ifname enc1e00
connection.autoconnect yes
Connection 'enc1e00' (f362f266-c68c-4b49-916d-950be6ffb64f) successfully added.

# nmcli connection modify enc1e00 ipv4.method manual ipv4.addresses
10.12.6.136/24

# nmcli con up enc1e00
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/3)

# nmcli connection show
NAME           UUID                                  TYPE      DEVICE
System enc640  8dd76dd1-ca0f-4e0a-a3e6-32492d2d7dac  ethernet  enc640
enc1e00        f362f266-c68c-4b49-916d-950be6ffb64f  ethernet  enc1e00

# nmcli connection show enc1e00 |grep ipv4.addresses
ipv4.addresses:                         10.12.6.136/24

# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enc640: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 02:01:02:00:00:2e brd ff:ff:ff:ff:ff:ff
    inet 129.40.23.136/24 brd 129.40.23.255 scope global noprefixroute enc640
       valid_lft forever preferred_lft forever
    inet6 fe80::1:2ff:fe00:2e/64 scope link
       valid_lft forever preferred_lft forever
3: enc1e00: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 02:01:02:00:00:2f brd ff:ff:ff:ff:ff:ff
    inet 10.12.6.136/24 brd 10.12.6.255 scope global noprefixroute enc1e00
       valid_lft forever preferred_lft forever
    inet6 fe80::5daa:fe72:3dc5:9857/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Repeat these steps for the rdbkrh13 server. Use information in Table 4-2 on page 72 to perform the network configuration on rdbkrh13 server.

SMC-R (RoCE)

In this section, we provide the basic commands to enable SMC-R on the Red Hat Enterprise Linux 8 server. SMC also can be enabled between different CPCs by using a RoCE card that allows remote direct memory access (RDMA) over the external network (SMC-R).

While connected to the RDBKRH05 server, complete the following steps:

1. Check the RoCE device availability by using the following command:

```
# lspci
00:00.0 Ethernet controller: Mellanox Technologies MT27710 Family [ConnectX-4
Lx Virtual Function]
```

> **Note:** lspci is available when you install the `pciutils` package.

2. Use the following command to check the PNET ID in the RoCE device:

```
# cat /sys/devices/pci0000\:00/0000\:00\:00.0/util_string | iconv -f IBM-1047
-t ASCII ; echo
PERFNET

# For OSA 1E00
# cat /sys/devices/css0/chp0.e2/util_string | iconv -f IBM-1047 -t ASCII ; echo
PERFNET
```

> **Note:** In the output that is shown in Step 2, the PNET ID in the OSA card is displayed. It also has same value of PCI device. Our OSA card for SMC was set up on CHPID E2.

3. Query the `smc-tools` package by using the following command to determine the full path for the `libsmc-preload` library. This library can be used if you have an application that uses SMC:

```
# rpm -ql smc-tools |grep libsmc
/usr/lib64/libsmc-preload.so
/usr/lib64/libsmc-preload.so.1

# This is the file we will use
# ls -la /usr/lib64/libsmc-preload.so
-rwsr-xr-x. 1 root root 13008 Nov  6  2019 /usr/lib64/libsmc-preload.so

# This is a soft link
# ls -la /usr/lib64/libsmc-preload.so.1
lrwxrwxrwx. 1 root root 17 Nov  6  2019 /usr/lib64/libsmc-preload.so.1 ->
libsmc-preload.so
```

4. To test the communication between two different LPARs in the same CPC that use SMC-R, we use the `iperf3` tool. To install it, run the following yum command:

```
# yum -y install iperf3
Updating Subscription Management repositories.
Unable to read consumer identity
This system is not registered to Red Hat Subscription Management. You can use
subscription-manager to register.
Last metadata expiration check: 0:10:07 ago on Tue 06 Oct 2020 11:22:50 AM EDT.
Dependencies resolved.
================================================================================
================================================
```

```
 Package                 Architecture            Version
Repository                                  Size
================================================================================
================================================
Installing:
 iperf3                  s390x                   3.5-3.el8
InstallMedia-AppStream                  89 k

Transaction Summary
================================================================================
================================================
Install  1 Package

Total size: 89 k
Installed size: 217 k
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
 Preparing     :
1/1
 Installing    : iperf3-3.5-3.el8.s390x
1/1
 Running scriptlet: iperf3-3.5-3.el8.s390x
1/1
 Verifying     : iperf3-3.5-3.el8.s390x
1/1
Installed products updated.

Installed:
 iperf3-3.5-3.el8.s390x

Complete!
```

5. Allow the local firewall to accept connections for **iperf3** on the 5201 TCP port:

```
# firewall-cmd --permanent --add-port=5201/tcp
success

# firewall-cmd --reload
success

# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enc1e00 enc640
  sources:
  services: cockpit dhcpv6-client ssh vnc-server
  ports: 5201/tcp
  protocols:
  masquerade: yes
  forward-ports:
  source-ports:
```

```
  icmp-blocks:
  rich rules:
```

6. Start **iperf3** in listening mode by using the following command:

```
# smc_run iperf3 -s
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

Note: smc_run starts a TCP socket program with the capability to use SMC as networking protocol

7. Use the following command to open another SSH session against the RDBKRH05 server and print out the information about the SMC sockets:

```
# smcss -a
State          UID   Inode   Local Address           Peer Address
Intf Mode
INIT           00000 0000000
LISTEN         00000 0026212 0.0.0.0:5201
```

Connect to the RDBKRH13 server and complete the following steps:

1. Check the RoCE device availability by using the following command:

```
# lspci
00:00.0 Ethernet controller: Mellanox Technologies MT27710 Family [ConnectX-4
Lx Virtual Function]
```

Note: The **lspci** command is available when you install the pciutils package.

2. To check the PNET ID in the RoCE device, use the following commands:

```
# cat /sys/devices/pci0000\:00/0000\:00\:00.0/util_string | iconv -f IBM-1047
-t ASCII ; echo
PERFNET

# For OSA 1E00
# cat /sys/devices/css0/chp0.e2/util_string | iconv -f IBM-1047 -t ASCII ; echo
PERFNET
```

Note: The PNET ID in both the OSA card and PCI device have same value.

3. Install the **iperf3** package by using yum:

```
yum -y install iperf3
```

4. To test SMC connections, issue the following **iperf3** command:

```
smc_run iperf3 -c 10.12.6.136 -t 60
Connecting to host 10.12.6.136, port 5201
[  5] local 10.12.6.149 port 49176 connected to 10.12.6.136 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.02   sec  1.90 GBytes  15.9 Gbits/sec    0   14.1 KBytes
[  5]   1.02-2.00   sec  1.49 GBytes  13.1 Gbits/sec    0   14.1 KBytes
[  5]   2.00-3.00   sec  1.67 GBytes  14.3 Gbits/sec    0   14.1 KBytes
```

```
[  5]   3.00-4.00   sec  1.83 GBytes  15.7 Gbits/sec    0   14.1 KBytes
[  5]   4.00-5.00   sec  1.70 GBytes  14.6 Gbits/sec    0   14.1 KBytes
[  5]   5.00-6.00   sec  1.57 GBytes  13.5 Gbits/sec    0   14.1 KBytes
[  5]   6.00-7.00   sec  1.65 GBytes  14.1 Gbits/sec    0   14.1 KBytes
[  5]   7.00-8.00   sec  1.83 GBytes  15.7 Gbits/sec    0   14.1 KBytes
[  5]   8.00-9.00   sec  2.20 GBytes  18.9 Gbits/sec    0   14.1 KBytes
[  5]   9.00-10.00  sec  1.69 GBytes  14.5 Gbits/sec    0   14.1 KBytes
[  5]  10.00-11.00  sec  1.23 GBytes  10.6 Gbits/sec    0   14.1 KBytes
[  5]  11.00-12.00  sec  1.35 GBytes  11.6 Gbits/sec    0   14.1 KBytes
[  5]  12.00-13.00  sec  2.03 GBytes  17.4 Gbits/sec    0   14.1 KBytes
[  5]  13.00-14.00  sec  1.84 GBytes  15.8 Gbits/sec    0   14.1 KBytes
[  5]  14.00-15.00  sec  1.98 GBytes  17.0 Gbits/sec    0   14.1 KBytes
[  5]  15.00-16.00  sec  1.92 GBytes  16.5 Gbits/sec    0   14.1 KBytes
[  5]  16.00-17.00  sec  1.68 GBytes  14.4 Gbits/sec    0   14.1 KBytes
[  5]  17.00-18.00  sec  1.89 GBytes  16.2 Gbits/sec    0   14.1 KBytes
[  5]  18.00-19.00  sec  1.74 GBytes  14.9 Gbits/sec    0   14.1 KBytes
[  5]  19.00-20.00  sec  1.76 GBytes  15.2 Gbits/sec    0   14.1 KBytes
[  5]  20.00-21.00  sec  1.82 GBytes  15.6 Gbits/sec    0   14.1 KBytes
.... OUTPUT OMMITED ....
```

Issue the command that is shown in Example 4-5 on the RDBKRH05 server SSH session to show the communication between two different servers that use SMC-R. The output that shown in Example 4-5 demonstrates the connections that use SMC-R.

Example 4-5 Show communications between two servers

```
# smcss -a
State          UID   Inode   Local Address           Peer Address            Intf
Mode
INIT           00000 0000000
ACTIVE         00000 0030879 ::ffff:10.12.6.136:5201 ::ffff:10.12.6...:49108 0000
SMCR
ACTIVE         00000 0030877 ::ffff:10.12.6.136:5201 ::ffff:10.12.6...:49106 0000
SMCR
LISTEN         00000 0030787 0.0.0.0:5201
```

For more information about RoCE, see the IBM Documentation.

Enabling SMC-D (ISM)

This section provides information about how to enable SMC-D.

In this section, we use same Red Hat Enterprise Linux servers that were used in “SMC-R (RoCE)” on page 79 to avoid redefining the OSA card and IP addresses.

Our ISM environment is described in Table 4-3.

Table 4-3 ISM environment

Server name	LPAR	ISM address	OSA CHPID	OSA Real Channels	(enc1e00) IP address
rdbkrh05	RDBKZVM2	00001031	E2	1E24-1E26	10.12.6.136
rdbkrh13	RDBKZVM1	00001030	E2	1E24-1E26	10.12.6.149

Complete the following steps on server RDBKRH05:

1. Connect to z/VM RDBKZVM2 by using an x3270 terminal.
2. Take down the RDBKRH05 server by using the following command:

   CP SIGNAL SHUTDOWN RDBKRH05 WITHIN 60

3. Issue the following **CP QUERY PCIF** command to get available ISM devices:

```
QUERY PCIF
08:58:47 PCIF 000000C2 FREE                          DISABLED RoCE Express 2
08:58:47 PCIF 000000D2 FREE                          DISABLED RoCE Express 2
08:58:47 PCIF 000000E2 FREE                          DISABLED RoCE Express 2
08:58:47 PCIF 000000F2 FREE                          DISABLED RoCE Express 2
08:58:47 PCIF 00001031 FREE                          DISABLED ISM
Ready; T=0.01/0.01 08:58:47
```

4. Update the RDBKRH05 server user directory by completing the following steps to include the ISM device:
   a. Remove any COMMAND ATTACH PCIF statement.
   b. Add the following ATTACH statement to the zVM user directory to attach 1031 PCI device:

      COMMAND ATTACH PCIF 1031 TO &USERID

5. Bring up the RDBKRH05 server up by using the **XAUTOLOG RDBKRH05** command.
6. Confirm the status of the ISM device by using the following command:

```
QUERY PCIF
09:01:14 PCIF 000000C2 FREE                          DISABLED RoCE Express 2
09:01:14 PCIF 000000D2 FREE                          DISABLED RoCE Express 2
09:01:14 PCIF 000000E2 FREE                          DISABLED RoCE Express 2
09:01:14 PCIF 000000F2 FREE                          DISABLED RoCE Express 2
09:01:14 PCIF 00001031 ATTACHED TO RDBKRH05 00001031 ENABLED  ISM
Ready; T=0.01/0.01 09:01:14
```

7. Confirm that the smc-tool is installed by using the **yum install -y sThe mc-tools** command.
8. Check the ISM device availability by using the following commands. The PNET ID in the OSA card is displayed as output from the **lspci** command:

```
# lspci
00:00.0 Non-VGA unclassified device: IBM Internal Shared Memory (ISM) virtual
PCI device
# vmcp q pcif
PCIF 00001031 ON 00001031 ENABLED  ISM
```

9. To check the PNET ID in the ISM device, use the following commands:

```
# cat /sys/devices/pci0000\:00/0000\:00\:00.0/util_string | iconv -f IBM-1047
-t ASCII ; echo
PERFNET

# cat /sys/devices/css0/chp0.e2/util_string | iconv -f IBM-1047 -t ASCII ; echo
PERFNET
```

10. Query SMC RoCE Express interfaces (rnics) by using the following command. The output shows that the interface column is listing N/A. Therefore, we remove the `encp0s0` profile and add the configuration onto `enc1e00`:

```
# smc_rnics
FID  Power  PCI ID        PCHID  Type           Port  PNET ID
Interface
--------------------------------------------------------------------------------
-----
1031  1      0000:00:00.0  07c0   ISM             n/a   PERFNET           n/a
```

11. Confirm that the local firewall accepts connections for **`iperf3`** on 5201 TCP port by using the following command:

```
# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enc1e00 enc640
  sources:
  services: cockpit dhcpv6-client ssh vnc-server
  ports: 5201/tcp
  protocols:
  masquerade: yes
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

12. Start **`iperf3`** in listening mode:

```
# smc_run iperf3 -s
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

> **Note:** The **`smc_run`** command starts a TCP socket program with the ability to use SMC as the networking protocol.

13. Open another SSH session to the `RDBKRH05` server and print the information about the SMC sockets by using the following command:

```
# smcss -a
State          UID   Inode   Local Address           Peer Address
Intf Mode
INIT           00000 0000000
LISTEN         00000 0026212 0.0.0.0:5201
```

Connect to the RDBKRH13 server and complete the following steps:

1. Check the ISM device availability by using the following command:

```
# lspci
00:00.0 Ethernet controller: Mellanox Technologies MT27710 Family [ConnectX-4
Lx Virtual Function]
```

2. To check the PNET ID in the ISM device, run the following command:

```
# cat /sys/devices/pci0000\:00/0000\:00\:00.0/util_string | iconv -f IBM-1047
-t ASCII ; echo
PERFNET
```

```
# For OSA 1E00
# cat /sys/devices/css0/chp0.e2/util_string | iconv -f IBM-1047 -t ASCII ; echo
PERFNET
```

Note: The PNET ID in the OSA card and the PCI device feature the same value.

3. Issue the command that is shown in Example 4-6 to test SMC-D connections.

Example 4-6 Test the SMC-D connections

```
smc_run iperf3 -c 10.12.6.136 -t 60
Connecting to host 10.12.6.136, port 5201
[  5] local 10.12.6.149 port 49176 connected to 10.12.6.136 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.02   sec  1.90 GBytes  15.9 Gbits/sec    0   14.1 KBytes
[  5]   1.02-2.00   sec  1.49 GBytes  13.1 Gbits/sec    0   14.1 KBytes
[  5]   2.00-3.00   sec  1.67 GBytes  14.3 Gbits/sec    0   14.1 KBytes
[  5]   3.00-4.00   sec  1.83 GBytes 15.7 Gbits/sec    0   14.1 KBytes
[  5]   4.00-5.00   sec  1.70 GBytes  14.6 Gbits/sec    0   14.1 KBytes
[  5]   5.00-6.00   sec  1.57 GBytes  13.5 Gbits/sec    0   14.1 KBytes
[  5]   6.00-7.00   sec  1.65 GBytes  14.1 Gbits/sec    0   14.1 KBytes
[  5]   7.00-8.00   sec  1.83 GBytes  15.7 Gbits/sec    0   14.1 KBytes
[  5]   8.00-9.00   sec  2.20 GBytes  18.9 Gbits/sec    0   14.1 KBytes
[  5]   9.00-10.00  sec  1.69 GBytes  14.5 Gbits/sec    0   14.1 KBytes
[  5]  10.00-11.00  sec  1.23 GBytes  10.6 Gbits/sec    0   14.1 KBytes
[  5]  11.00-12.00  sec  1.35 GBytes  11.6 Gbits/sec    0   14.1 KBytes
[  5]  12.00-13.00  sec  2.03 GBytes  17.4 Gbits/sec    0   14.1 KBytes
[  5]  13.00-14.00  sec  1.84 GBytes  15.8 Gbits/sec    0   14.1 KBytes
[  5]  14.00-15.00  sec  1.98 GBytes  17.0 Gbits/sec    0   14.1 KBytes
[  5]  15.00-16.00  sec  1.92 GBytes  16.5 Gbits/sec    0   14.1 KBytes
[  5]  16.00-17.00  sec  1.68 GBytes  14.4 Gbits/sec    0   14.1 KBytes
[  5]  17.00-18.00  sec  1.89 GBytes  16.2 Gbits/sec    0   14.1 KBytes
[  5]  18.00-19.00  sec  1.74 GBytes  14.9 Gbits/sec    0   14.1 KBytes
[  5]  19.00-20.00  sec  1.76 GBytes  15.2 Gbits/sec    0   14.1 KBytes
[  5]  20.00-21.00  sec  1.82 GBytes  15.6 Gbits/sec    0   14.1 KBytes
... OUTPUT OMMITED ...
```

4. Return to the RDBKRH05 SSH session and issue the following command to show the communication between the two different servers that are using SMC-D:

```
# smcss -a
State          UID   Inode   Local Address           Peer Address
Intf Mode
INIT           00000 0000000
ACTIVE         00000 0026225 ::ffff:10.12.6.136:5201 ::ffff:10.12.6...:49176
0000 SMCD
ACTIVE         00000 0026223 ::ffff:10.12.6.136:5201 ::ffff:10.12.6...:49174
0000 SMCD
LISTEN         00000 0026212 0.0.0.0:5201
```

Considerations

During our tests, we found problems where some kernel modules were not automatically loaded after reboot and this issue prevented SMCD and SMCR from working correctly. We ran the following command to verify that our output showed Mode = TCP:

```
# smcss -a
State          UID   Inode   Local Address           Peer Address            Intf
Mode
INIT           00000 0000000
ACTIVE         00000 0027073 ::ffff:10.12.6.136:5201 ::ffff:10.12.6...:43782 0000
TCP 0x05000000/0x03030000
ACTIVE         00000 0027071 ::ffff:10.12.6.136:5201 ::ffff:10.12.6...:43780 0000
TCP 0x05000000/0x03030000
LISTEN         00000 0027067 0.0.0.0:5201
```

We next verified if the modules **smc, mlx5**, and **ib_core** were loaded by using the command that is shown in Example 4-7. If they are not loaded, use the **modprobe** command to load them and then, retry the tests that are shown in Example 4-5 on page 82 and Example 4-6 on page 85.

Example 4-7 Verifying that modules are loaded

```
# lsmod|egrep "smc|mlx5|ib_core"
smc_diag               16384  0
smc                   110592  4 smc_diag
pnet                   16384  1 smc
mlx5_ib               299008  0
ib_uverbs             126976  1 mlx5_ib
ib_core               352256  3 smc,ib_uverbs,mlx5_ib
mlx5_core             987136  1 mlx5_ib
mlxfw                  20480  1 mlx5_core
```

4.8 Working with HiperSockets

HiperSockets allows memory-to-memory communication between hosts in the same CPC. HiperSockets avoids the use of external communications by way of NIC and Ethernet switch. This feature eliminates traditional network latency because no hardware is required.

4.8.1 Setting up a HiperSockets network

To set up a HiperSockets network, complete the following steps:

1. “Defining and attaching the devices” on page 87.
2. “Defining network configuration on Red Hat Enterprise Linux servers” on page 89.

In this section, we define HiperSockets for the systems that are listed in Table 4-4.

Table 4-4 Our Red Hat environment

LPAR name	Linux VM ID	CHPID	Real device	Virtual device	IP address
ARIES32	RDBKRH04	F0	0FA0-0FA2	7000-7002	192.168.1.7
ARIES31	RDBKRH11	F0	0FA0-0FA2	7000-7002	192.168.1.8

Note: Real devices 0FA0-0FA2 are shared between ARIES31 and ARIES32.

Defining and attaching the devices

An I/O configuration can be built or updating by using one of the following methods:

- Hardware Configuration Definition (HCD): A panel-driven ISPF application.
- Hardware Configuration Manager (HCM): A workstation-based GUI.
- Manually code the IOCP statements.

In all three cases, it is the I/O Configuration Program (IOCP), which is called under the covers by the HCD and the HCM, that processes the IOCP statements and builds an IODF. The IODF is then used to update the active system and an IOCDS slot.

To run the IOCP, specify the logical partition, the channel paths on the CPC, and how they are assigned to the logical partitions, the control units that are attached to the channel paths, and the devices that are assigned to the control units.

In our lab environment, we chose to manually code the IOCP statements by using the following steps:

1. Define the channel path.

   You can use the definition that is shown in Example 4-8 to create the statements for the channel path (CHPID) to use with a HiperSockets connection. You define the CHPID, control unit, and devices.

   Example 4-8 IOCP definition

```
CHPID PATH=(F0),PART=((ARIES31,ARIES32)),CHPARM=40,TYPE=IQD
CNTLUNIT CUNUMBR=F000,PATH=(F0),UNIT=IQD?IODEVICE
ADDRESS=(F000,12),CUNUMBR=F000,UNIT=IQD
```

   Where:

   FE is the path address
   ARIES31 and **ARIES32** are the names of the partitions that have access to the HiperSocket at Power on Reset (POR). The CHPID is defined implicitly as shared.

   FE00 is an available device number. Its value was selected to clarify its relationship to the CHPID number.
   CHPARM=40 defines the HiperSocket frame size as 24KB, enabling use of a maximum transmission unit (MTU) value up to 16KB (16 384 bytes). An MTU value of 8992 enables TCP/IP to forward jumbo Ethernet frames across the HiperSocket interface without fragmentation. The CHPARM value may be selected depending on the largest MTU you require.

   In our environment, the IBM Z configuration team created the following IOCP configuration (asterisk [*] must be in column 72).

```
      CHPID PATH=(CSS(0,1,2,3),F0),SHARED,                          *
               NOTPART=((CSS(0),(ARIESOA,ARIESOB),(=)),(CSS(2),(ARIES2C*
               ,ARIES2D,ARIES2E,ARIES2F),(=))),VCHID=7E0,TYPE=IQD
         CHPID PATH=(CSS(0,1,2,3),F1),SHARED,                          *
               NOTPART=((CSS(0),(ARIESOA,ARIESOB),(=)),(CSS(2),(ARIES2C*
               ,ARIES2D,ARIES2E,ARIES2F),(=))),VCHID=7E1,TYPE=IQD
         CHPID PATH=(CSS(0,1,2,3),F2),SHARED,                          *
               NOTPART=((CSS(0),(ARIESOA,ARIESOB),(=)),(CSS(2),(ARIES2C*
```

```
      ,ARIES2D,ARIES2E,ARIES2F),(=))),CHPARM=04,VCHID=7E2,    *
      PNETID=PERFNET,TYPE=IQD
CHPID PATH=(CSS(0,1,2,3),F4),SHARED,                          *
      NOTPART=((CSS(0),(ARIESOA,ARIESOB),(=)),(CSS(2),(ARIES2C*
      ,ARIES2D,ARIES2E,ARIES2F),(=))),VCHID=7E4,TYPE=IQD
CHPID PATH=(CSS(0,1,2,3),F5),SHARED,                          *
      NOTPART=((CSS(0),(ARIESOA,ARIESOB),(=)),(CSS(2),(ARIES2C*
      ,ARIES2D,ARIES2E,ARIES2F),(=))),VCHID=7E5,              *
      PNETID=REDBOOKS,TYPE=IQD
CNTLUNIT CUNUMBR=F000,                                        *
      PATH=((CSS(0),F0),(CSS(1),F0),(CSS(2),F0),(CSS(3),F0)), *
      UNIT=IQD
IODEVICE ADDRESS=(FA0,031),UNITADD=00,CUNUMBR=(F000),UNIT=IQD
CNTLUNIT CUNUMBR=F100,                                        *
      PATH=((CSS(0),F1),(CSS(1),F1),(CSS(2),F1),(CSS(3),F1)), *
      UNIT=IQD
IODEVICE ADDRESS=(FE0,031),UNITADD=00,CUNUMBR=(F100),UNIT=IQD
CNTLUNIT CUNUMBR=F200,                                        *
      PATH=((CSS(0),F2),(CSS(1),F2),(CSS(2),F2),(CSS(3),F2)), *
      UNIT=IQD
IODEVICE ADDRESS=(FC0,032),UNITADD=00,CUNUMBR=(F200),UNIT=IQD
CNTLUNIT CUNUMBR=F400,                                        *
      PATH=((CSS(0),F4),(CSS(1),F4),(CSS(2),F4),(CSS(3),F4)), *
      UNIT=IQD
IODEVICE ADDRESS=(F00,032),CUNUMBR=(F400),UNIT=IQD
CNTLUNIT CUNUMBR=F500,                                        *
      PATH=((CSS(0),F5),(CSS(1),F5),(CSS(2),F5),(CSS(3),F5)), *
      UNIT=IQD
IODEVICE ADDRESS=(F20,032),UNITADD=00,CUNUMBR=(F500),UNIT=IQD
```

2. Attach the devices.

   Before attaching the devices, you must verify the Channel Path.

   For an IOCP verification, use the following CP command that is connected in your zVM LPAR to verify that the channel path to each device is online to the guest systems, where F0 is the CHPID number:

   ```
   QUERY CHPID F0
   07:25:46 Path F0 online to devices 0FA0 0FA1 0FA2 0FA3 0FA4 0FA5 0FA6 0FA7
   07:25:46 Path F0 online to devices 0FA8 0FA9 0FAA 0FAB 0FAC 0FAD 0FAE 0FAF
   07:25:46 Path F0 online to devices 0FB0 0FB1 0FB2 0FB3 0FB4 0FB5 0FB6 0FB7
   07:25:46 Path F0 online to devices 0FB8 0FB9 0FBA 0FBB 0FBC 0FBD 0FBE
   ```

   You received PATH messages that indicate the devices are online.

   Use the following command to verify device allocation:

   ```
   QUERY 0FA0 0FA1 0FA2
   07:56:59 OSA  0FA0 FREE    , OSA  0FA1 FREE    , OSA  0FA2 FREE
   ```

   Optionally, you can confirm if Linux servers are located in the same CPC, by using the following commands that are connected to the Red Hat servers by way of SSH:

   ```
   On rdbkrh04:
   # grep Sequence /proc/sysinfo
   Sequence Code:        000000000002B7F8

   On rdbkrh11:
   # grep Sequence /proc/sysinfo
   ```

```
Sequence Code:         000000000002B7F8
```

You can get the LPAR name by using the following commands by way of an SSH session:

```
On rdbkrh04:
# grep "^LPAR Name" /proc/sysinfo
LPAR Name:             ARIES32

On rdbkrh11:
# grep "^LPAR Name" /proc/sysinfo
LPAR Name:             ARIES31
```

On the zVM systems RDBKZVM1 and RDBKZVM2, complete the following steps:

1. Connect to RDBKZVM1 as MAINT.
2. Attach the first three devices to RDBKRH04:

```
ATTACH 0FA0 RDBKRH04 7000
07:57:14 OSA 0FA0 ATTACHED TO RDBKRH04 7000
Ready; T=0.01/0.01 07:57:14

ATTACH 0FA1 RDBKRH04 7001
07:58:16 OSA 0FA1 ATTACHED TO RDBKRH04 7001
Ready; T=0.01/0.01 07:58:16

ATTACH 0FA2 RDBKRH04 7002
07:58:24 OSA 0FA2 ATTACHED TO RDBKRH04 7002
Ready; T=0.01/0.01 07:58:24
```

3. To attach the HiperSockets I/O devices to a virtual machine when it is started, place DEDICATE or COMMAND ATTACH statements in the virtual machine's directory entry. Each HiperSockets interface requires three consecutive I/O devices on the same HiperSocket CHPID:

```
DIRM FOR RDBKRH04 DEDICATE 7000 0FA0
DIRM FOR RDBKRH04 DEDICATE 7001 0FA1
DIRM FOR RDBKRH04 DEDICATE 7002 0FA2
```

Defining network configuration on Red Hat Enterprise Linux servers

On the Red Hat Enterprise Linux server (RDBKRH04), complete the following steps:

1. Issue the **lscss** command to display network devices:

```
# vmcp q 7000-7002
OSA  7000 ON OSA   0FA0 SUBCHANNEL = 001E
     7000 DEVTYPE HIPER       VIRTUAL CHPID F0 IQD REAL CHPID F0
     7000 QDIO-ELIGIBLE       QIOASSIST-ELIGIBLE
OSA  7001 ON OSA   0FA1 SUBCHANNEL = 001F
     7001 DEVTYPE HIPER       VIRTUAL CHPID F0 IQD REAL CHPID F0
     7001 QDIO-ELIGIBLE       QIOASSIST-ELIGIBLE
OSA  7002 ON OSA   0FA2 SUBCHANNEL = 0020
     7002 DEVTYPE HIPER       VIRTUAL CHPID F0 IQD REAL CHPID F0
     7002 QDIO-ELIGIBLE       QIOASSIST-ELIGIBLE
```

2. On an RHEL system, the HiperSockets devices might not appear, although they are configured correctly in the IOCDS and in z/VM. In this case, to remove the HiperSockets devices from the list of ignored devices and make them visible to Linux, issue the following command:

```
# cio_ignore -r 0.0.7000,0.0.7001,0.0.7002
```

3. Issue **lscss** command to display our network devices (highlighted in bold):

```
# lscss | grep 1732
0.0.0640 0.0.0006  1732/01 1731/01 yes  80  80  ff   02000000 00000000
0.0.0641 0.0.0007  1732/01 1731/01 yes  80  80  ff   02000000 00000000
0.0.0642 0.0.0008  1732/01 1731/01 yes  80  80  ff   02000000 00000000
0.0.b743 0.0.001c  1732/03 1731/03 yes  80  80  ff   25000000 00000000
0.0.c743 0.0.001d  1732/03 1731/03 yes  80  80  ff   35000000 00000000
0.0.7000 0.0.001e  1732/05 1731/05      80  80  ff   f0000000 00000000
0.0.7001 0.0.001f  1732/05 1731/05      80  80  ff   f0000000 00000000
0.0.7002 0.0.0020  1732/05 1731/05      80  80  ff   f0000000 00000000
```

This output shows that the Linux system knows of the devices, but the absence of the word yes between the 1731/05 and the 80 indicates that the devices are not configured or online. The first two characters of the second to the last column show the CHPID, f0, in the example.

4. Use the **chzdev** command to configure and **lszdev** to list configurations for network devices. The following example shows how to use the **chzdev** command to set up the HiperSockets device and **lszdev** to list configured network device:

```
# chzdev -e qeth 0.0.7000,0.0.7001,0.0.7002
QETH device 0.0.7000:0.0.7001:0.0.7002 configured

# lszdev qeth 7000
TYPE  ID                          ON   PERS  NAMES
qeth  0.0.7000:0.0.7001:0.0.7002  yes  yes   enc7000
```

Verifying the setup

This section describes how to verify the configuration, regardless of the way that it is defined. The following example shows how the **lsqeth** command can be used to display the attributes of an interface:

```
# lsqeth enc7000
Device name                    : enc7000
------------------------------------------------------------------------------
        card_type              : HiperSockets
        cdev0                  : 0.0.7000
        cdev1                  : 0.0.7001
        cdev2                  : 0.0.7002
        chpid                  : F0
        online                 : 1
        portname               : no portname required
        portno                 : 0
        route4                 : no
        route6                 : no
        state                  : UP (LAN ONLINE)
        priority_queueing      : always queue 2
        fake_broadcast         : 0
        buffer_count           : 128
        layer2                 : 0
        isolation              : none
        sniffer                : 0
```

The following example shows how to set the new HiperSockets device and assign an IP address:

```
# nmcli device
DEVICE   TYPE      STATE         CONNECTION
enc640   ethernet  connected     System enc640
enc7000  ethernet  disconnected  --
lo       loopback  unmanaged     --

# nmcli connection add type ethernet con-name enc7000 connection.autoconnect
yes
Connection 'enc7000' (ce9d4fc6-7ebd-4f4e-9ad9-ae0dbf9e41e1) successfully added.

# nmcli connection modify enc7000 ipv4.method manual ipv4.addresses
192.168.1.7/24

# nmcli con up enc7000
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/7)

# nmcli con show
NAME           UUID                                   TYPE      DEVICE
System enc640  9ff8262c-df27-4d5e-9741-01e0cafde28f  ethernet  enc640
enc7000        9c3bbc86-dfbd-4d92-9668-3a6413f96503  ethernet  enc7000

# nmcli device
DEVICE   TYPE      STATE      CONNECTION
enc640   ethernet  connected  System enc640
enc7000  ethernet  connected  enc7000
lo       loopback  unmanaged  --

# nmcli device
DEVICE   TYPE      STATE      CONNECTION
enc640   ethernet  connected  System enc640
enc7000  ethernet  connected  enc7000
lo       loopback  unmanaged  --

# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enc640: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 02:01:02:00:00:09 brd ff:ff:ff:ff:ff:ff
    inet 129.40.23.135/24 brd 129.40.23.255 scope global noprefixroute enc640
       valid_lft forever preferred_lft forever
    inet6 fe80::1:2ff:fe00:9/64 scope link
       valid_lft forever preferred_lft forever
3: enc7000: <BROADCAST,MULTICAST,NOARP,UP,LOWER_UP> mtu 8192 qdisc mq state UP
group default qlen 1000
    link/ether 0e:00:f0:32:00:02 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.7/24 brd 192.168.1.255 scope global noprefixroute enc7000
```

```
       valid_lft forever preferred_lft forever
    inet6 fe80::70d9:a232:6fd6:9c63/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

# ping -c2 192.168.1.7
PING 192.168.1.7 (192.168.1.7) 56(84) bytes of data.
64 bytes from 192.168.1.7: icmp_seq=1 ttl=64 time=0.046 ms
64 bytes from 192.168.1.7: icmp_seq=2 ttl=64 time=0.084 ms

--- 192.168.1.7 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 52ms
rtt min/avg/max/mdev = 0.046/0.065/0.084/0.019 ms
```

Repeat these steps for `RDBKRH11` and `RDBKZVM1` LPAR. Remember to use the correct IP address.

After completing the configuration for the RDBKRH11 server, attempt to ping the other server by using the HiperSockets' IP address:

```
On rdbkrh04 server:
# ping -c3 192.168.1.8
PING 192.168.1.8 (192.168.1.8) 56(84) bytes of data.
64 bytes from 192.168.1.8: icmp_seq=1 ttl=64 time=0.697 ms
64 bytes from 192.168.1.8: icmp_seq=2 ttl=64 time=0.431 ms
64 bytes from 192.168.1.8: icmp_seq=3 ttl=64 time=2.74 ms

--- 192.168.1.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 53ms
rtt min/avg/max/mdev = 0.431/1.288/2.737/1.030 ms

On rdbkrh11 server:
# ping -c3 192.168.1.7
PING 192.168.1.7 (192.168.1.7) 56(84) bytes of data.
64 bytes from 192.168.1.7: icmp_seq=1 ttl=64 time=0.197 ms
64 bytes from 192.168.1.7: icmp_seq=2 ttl=64 time=0.368 ms
64 bytes from 192.168.1.7: icmp_seq=3 ttl=64 time=0.380 ms

--- 192.168.1.7 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 114ms
rtt min/avg/max/mdev = 0.197/0.315/0.380/0.083 ms
```

Part 2

Other topics

After you complete the installation of Red Hat Enterprise Linux, it is important to learn more about other tasks. This part of the book discusses those tasks and includes the following topics:

5

Working with disks

This chapter discusses working with disks. This chapter describes extended count key data (IBM ECKD) direct access storage device (DASD), and Fibre Channel Protocol (FCP)/Small Computer System Interface (SCSI) tasks that you might want to perform. It concentrates on the tasks to perform on Linux.

For more information about the z/VM perspective, see 11.2, “Considerations for disk storage types” in Volume 1 of this series in *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147.

This chapter includes the following topics:

5.1 Adding disk space to virtual machines

This section describes how to add disk space to a Linux virtual machine. This disk space can come from different types of disks. The types of disks are described in 11.2, “Considerations for disk storage types”, in Volume 1 of this series: *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147.

Important: If you add minidisks or add them to the user directory for a specific virtual machine, you can attach them to a running Linux system without “bouncing” (restarting) it.

For example, if you added a minidisk at virtual address 104, you can use the following commands to link to the disk and then enable it:

```
# vmcp link '* 104 104 mr' #or vmcp link \* 104 104 mr
# chccwdev -e 104
```

The asterisk (*) character on the **vmcp link** command refers to the current VM user ID.

Before you start working with disks for your Red Hat Enterprise Linux on IBM Z guest, you must define all necessary disks in the zVM user directory. In our environment, we ran the following DIRMAINT commands to add 102, 103, and 104 DASD minidisks for our Red Hat Enterprise Linux server (RDBKRH04). Issue the following commands connected as MAINT in your x3270 terminal:

```
DIRM FOR RDBKRH04 AMDISK 102 3390 AUTOG 10017 DASDBIG  MR
DIRM FOR RDBKRH04 AMDISK 103 3390 AUTOG 10017 DASDBIG  MR
DIRM FOR RDBKRH04 AMDISK 104 3390 AUTOG 10017 DASDBIG  MR
```

If your Red Hat Enterprise Linux server is up and running, issue the following commands to attach the new disks to your server:

```
vmcp link \* 102 102 mr
vmcp link \* 103 103 mr
vmcp link \* 104 104 mr
```

5.1.1 Making new minidisks or CKD DASD available

After obtaining new minidisks or count key data (CKD) DASD, make the new disks available (in our example we use addresses 0.0.0102, 0.0.0103, and 0.0.0104) by completing the following steps:

1. Check whether the disks are attached to the server. Issue the **vmcp q dasd** command as root. (The disks we are interested in are highlighted in bold.)

```
# vmcp q dasd
DASD 0100 3390 RDLXBQ R/W      89040 CYL ON DASD  90C1 SUBCHANNEL = 000A
DASD 0102 3390 RDLXB3 R/W      10017 CYL ON DASD  91B5 SUBCHANNEL = 0018
DASD 0103 3390 RDLXB4 R/W      10017 CYL ON DASD  90B6 SUBCHANNEL = 0019
DASD 0104 3390 RDLXB5 R/W      10017 CYL ON DASD  91B6 SUBCHANNEL = 001A
DASD 0190 3390 RD2RES R/O        214 CYL ON DASD  96B1 SUBCHANNEL = 000F
DASD 0191 3390 RDLX01 R/W          2 CYL ON DASD  99CF SUBCHANNEL = 0009
DASD 019D 3390 RD2RES R/O        292 CYL ON DASD  96B1 SUBCHANNEL = 0010
DASD 019E 3390 RD2RES R/O        500 CYL ON DASD  96B1 SUBCHANNEL = 0011
DASD 019F 3390 RD2U01 R/O        800 CYL ON DASD  9731 SUBCHANNEL = 0012
DASD 0300 9336 (VDSK) R/W     524288 BLK ON DASD  VDSK SUBCHANNEL = 0016
DASD 0301 9336 (VDSK) R/W    1048576 BLK ON DASD  VDSK SUBCHANNEL = 0017
```

```
DASD 0401 3390 RD2RES R/O        292 CYL ON DASD  96B1 SUBCHANNEL = 0014
DASD 0402 3390 RD2RES R/O        292 CYL ON DASD  96B1 SUBCHANNEL = 0013
DASD 0592 3390 RD2RES R/O        240 CYL ON DASD  96B1 SUBCHANNEL = 0015
```

2. Make the disks visible by using the **cio_ignore** command:

```
# cio_ignore -r 102
# cio_ignore -r 103
# cio_ignore -r 104
```

3. Enable the disks by using the **chccwdev -e** command:

```
# chccwdev -e 102 103 104
Setting device 0.0.0102 online
Done
Setting device 0.0.0103 online
Done
Setting device 0.0.0104 online
Done
```

4. Confirm whether disks are active and available by running the following command:

```
# lsdasd
Bus-ID    Status    Name      Device  Type         BlkSz  Size      Blocks
==============================================================================
=
0.0.0100  active    dasda     94:0    ECKD         4096   62606MB   16027200
0.0.0300  active    dasdb     94:4    FBA          512    256MB     524288
0.0.0301  active    dasdc     94:8    FBA          512    512MB     1048576
0.0.0102  n/f       dasdd     94:12   ECKD
0.0.0103  n/f       dasde     94:16   ECKD
0.0.0104  n/f       dasdf     94:20   ECKD
```

Note: The **lsdasd** command lists the disk as n/f (non-formatted).

5. Use the **dasdfmt** command to perform a low-level format as shown in Example 5-1.

Example 5-1 Format disks

```
dasdfmt -y -b 4096 -v /dev/disk/by-path/ccw-0.0.0102
   #you could use /dev/dasdd
dasdfmt -y -b 4096 -v /dev/disk/by-path/ccw-0.0.0103
   # you could use /dev/dasde
dasdfmt -y -b 4096 -v /dev/disk/by-path/ccw-0.0.0104
   # you could use /dev/dasdf
```

You should receive the output that is shown in Example 5-2.

Example 5-2 Results of format commands

```
# dasdfmt -y -b 4096 -v /dev/disk/by-path/ccw-0.0.0102
Retrieving disk geometry...
Drive Geometry: 10017 Cylinders * 15 Heads =  150255 Tracks
Device Type: Fully Provisioned

I am going to format the device /dev/disk/by-path/ccw-0.0.0102 in the following
way:
   Device number of device : 0x102
   Labelling device        : yes
```

```
   Disk label              : VOL1
   Disk identifier         : 0X0102
   Extent start (trk no)   : 0
   Extent end (trk no)     : 150254
   Compatible Disk Layout  : yes
   Blocksize               : 4096
   Mode                    : Full
Formatting the device. This may take a while (get yourself a coffee).
Detaching the device...
Invalidate first track...

formatting tracks complete...
Revalidate first track...
Reaccessing the device...
Finished formatting the device.
Retrieving dasd information... ok
Writing empty bootstrap...
Writing label...
Writing VTOC... ok
Rereading the partition table... ok

# dasdfmt -y -b 4096 -v /dev/disk/by-path/ccw-0.0.0103
Retrieving disk geometry...
Drive Geometry: 10017 Cylinders * 15 Heads =  150255 Tracks
Device Type: Fully Provisioned

I am going to format the device /dev/disk/by-path/ccw-0.0.0103 in the following
way:
   Device number of device : 0x103
   Labelling device        : yes
   Disk label              : VOL1
   Disk identifier         : 0X0103
   Extent start (trk no)   : 0
   Extent end (trk no)     : 150254
   Compatible Disk Layout  : yes
   Blocksize               : 4096
   Mode                    : Full
Formatting the device. This may take a while (get yourself a coffee).
Detaching the device...
Invalidate first track...

formatting tracks complete...
Revalidate first track...
Re-accessing the device...
Finished formatting the device.
Retrieving dasd information... ok
Writing empty bootstrap...
Writing label...
Writing VTOC... ok
Rereading the partition table... ok

# dasdfmt -y -b 4096 -v /dev/disk/by-path/ccw-0.0.0104
Retrieving disk geometry...
Drive Geometry: 10017 Cylinders * 15 Heads =  150255 Tracks
Device Type: Fully Provisioned
```

```
I am going to format the device /dev/disk/by-path/ccw-0.0.0104 in the following
way:
   Device number of device : 0x104
   Labelling device        : yes
   Disk label              : VOL1
   Disk identifier         : 0X0104
   Extent start (trk no)   : 0
   Extent end (trk no)     : 150254
   Compatible Disk Layout  : yes
   Blocksize               : 4096
   Mode                    : Full
Formatting the device. This may take a while (get yourself a coffee).
Detaching the device...
Invalidate first track...

formatting tracks complete...
Revalidate first track...
Reaccessing the device...
Finished formatting the device.
Retrieving dasd information... ok
Writing empty bootstrap...
Writing label...
Writing VTOC... ok
Rereading the partition table... ok
```

6. We used the command, **fdasd**, to partition the DASD. We created one partition that spanned the whole disk. Example 5-3 shows the commands we used and their output.

Example 5-3 Partitioning the disks with the output

```
fdasd -a /dev/disk/by-path/ccw-0.0.0102
   reading volume label ..: VOL1
   reading vtoc ..........: ok
   auto-creating one partition for the whole disk...
   writing volume label...
   writing VTOC...
   rereading partition table...
fdasd -a /dev/disk/by-path/ccw-0.0.0103
   reading volume label ..: VOL1
   reading vtoc ..........: ok
   auto-creating one partition for the whole disk...
   writing volume label...
   writing VTOC...
   rereading partition table...
fdasd -a /dev/disk/by-path/ccw-0.0.0104
   reading volume label ..: VOL1
   reading vtoc ..........: ok
   auto-creating one partition for the whole disk...
   writing volume label...
   writing VTOC...
   rereading partition table...
```

7. Run the **lsdasd** command to check status of the disks, as shown in the following example.

```
# lsdasd
Bus-ID    Status    Name      Device  Type         BlkSz  Size      Blocks
==============================================================================
=
0.0.0100  active    dasda     94:0    ECKD         4096   62606MB   16027200
0.0.0300  active    dasdb     94:4    FBA          512    256MB     524288
0.0.0301  active    dasdc     94:8    FBA          512    512MB     1048576
0.0.0102  active    dasdd     94:12   ECKD         4096   7043MB    1803060
0.0.0103  active    dasde     94:16   ECKD         4096   7043MB    1803060
0.0.0104  active    dasdf     94:20   ECKD         4096   7043MB    1803060
```

The disks are now formatted and can be used.

8. Make a backup of the configuration file, `/etc/dasd.conf`, and then, add minidisks `102`, `103`, and `104` to it by running the commands that are shown in Example 5-4.

Example 5-4 Backup configuration and add minidisks

```
# cd /etc
# cp dasd.conf dasd.conf.orig
# vi dasd.conf
   0.0.0100
   0.0.0300
   0.0.0301
   0.0.0102
   0.0.0103
   0.0.0104
```

Use the `/etc/dasd.conf` file for DASD disks that are not part of the root file system to make them persistent during reboots. This file contains one disk of DASD per line.

5.1.2 Making a new zFCP LUN available

To use Fibre Channel Protocol (FCP) in a single system image (SSI) environment, more adapters are available in Linux to be handled than are visible in one SSI node. The behavior of FCP to automatically detect the logical unit number (LUN) changed since Red Hat Enterprise Linux 7.1. Therefore, it is sufficient to configure the host adapters and use the multipathed device for disk configurations only.

This section assumes that no previous zFCP was available. The planning according to this book creates two FCP adapters at the addresses `0.0.b743` and `0.0.c743`. The necessary setup for z/VM is described in 11.2.2, “Direct-attached Fibre Channel”, in *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147.

In our lab environment, we ran the command that is shown in Example 5-5 to query the available FCP channels and to attach two FCP devices to our Red Hat Enterprise Linux server (RDBKRH04). Use these commands in your x3270 terminal and logged with a user with privilege B.

Example 5-5 Query available FCP channels

```
q fcp free
12:18:02 FCP  B740 FREE    , FCP  B741 FREE    , FCP  B742 FREE    , FCP  B743
FREE
12:18:02 FCP  B744 FREE    , FCP  B745 FREE    , FCP  B746 FREE    , FCP  B747
FREE
```

```
   12:18:02 FCP  B748 FREE    , FCP  B749 FREE    , FCP  B74A FREE    , FCP  B74B
   FREE
   12:18:02 FCP  B74C FREE    , FCP  B74D FREE    , FCP  B74E FREE    , FCP  B74F
   FREE
   12:18:02 FCP  B750 FREE    , FCP  B751 FREE    , FCP  B752 FREE    , FCP  B753
   FREE
   12:18:02 FCP  B754 FREE    , FCP  B755 FREE    , FCP  B756 FREE    , FCP  B757
   FREE
   12:18:02 FCP  B758 FREE    , FCP  B759 FREE    , FCP  B75A FREE    , FCP  B75B
   FREE
   12:18:02 FCP  B75C FREE    , FCP  B75D FREE    , FCP  B75E FREE    , FCP  B75F
   FREE
   12:18:02 FCP  C740 FREE    , FCP  C741 FREE    , FCP  C742 FREE    , FCP  C743
   FREE
   12:18:02 FCP  C744 FREE    , FCP  C745 FREE    , FCP  C746 FREE    , FCP  C747
   FREE
   12:18:02 FCP  C748 FREE    , FCP  C749 FREE    , FCP  C74A FREE    , FCP  C74B
   FREE
   12:18:02 FCP  C74C FREE    , FCP  C74D FREE    , FCP  C74E FREE    , FCP  C74F
   FREE
   12:18:02 FCP  C750 FREE    , FCP  C751 FREE    , FCP  C752 FREE    , FCP  C753
   FREE
   12:18:02 FCP  C754 FREE    , FCP  C755 FREE    , FCP  C756 FREE    , FCP  C757
   FREE
   12:18:02 FCP  C758 FREE    , FCP  C759 FREE    , FCP  C75A FREE    , FCP  C75B
   FREE
   12:18:02 FCP  C75C FREE    , FCP  C75D FREE    , FCP  C75E FREE    , FCP  C75F
   FREE
Ready; T=0.01/0.01 12:18:02
```

As shown in Example 5-5 on page 100, we attached b743 and c743 devices by using the following configuration:

```
attach b743 RDBKRH04
12:20:56 FCP B743 ATTACHED TO RDBKRH04 B743
Ready; T=0.01/0.01 12:20:56

attach c743 RDBKRH04
12:21:01 FCP C743 ATTACHED TO RDBKRH04 C743
Ready; T=0.01/0.01 12:21:01
```

To make a new FCP LUN available in Red Hat Enterprise Linux, complete the following steps on your Red Hat Enterprise Linux server:

1. Start a Secure Shell (SSH) session to the target system.
2. Check that two devices are available by using the **vmcp query fcp** command, as shown in Example 5-6. The device number and WWPN addresses are highlighted in bold.

Example 5-6 Check available devices

```
# vmcp q fcp
FCP  B743 ON FCP   B743 CHPID 25 SUBCHANNEL = 001C
     B743 DEVTYPE FCP         VIRTUAL CHPID 25 FCP REAL CHPID 25
     B743 QDIO-ELIGIBLE       QIOASSIST-ELIGIBLE
     B743 DATA ROUTER ELIGIBLE
     WWPN C05076D08002C40C
```

```
FCP  C743 ON FCP   C743 CHPID 35 SUBCHANNEL = 001D
     C743 DEVTYPE FCP         VIRTUAL CHPID 35 FCP REAL CHPID 35
     C743 QDIO-ELIGIBLE       QIOASSIST-ELIGIBLE
     C743 DATA ROUTER ELIGIBLE
     WWPN C05076D08002C60C
```

3. Make the disks visible by using the following **cio_ignore** command:

```
# cio_ignore -r b743
# cio_ignore -r c743
```

4. Enable the FCP adapters by using the following **chccwdev** command:

```
# chccwdev -e b743
Setting device 0.0.b743 online
Done
# chccwdev -e c743
Setting device 0.0.c743 online
Done
```

5. Verify that the auto LUN scan feature detected all of the paths to the LUNs by using the following command:

```
# lsluns
Scanning for LUNs on adapter 0.0.b743
   at port 0x5005076309141145:
      0x4001400300000000
   at port 0x50050763091b1145:
      0x4001400300000000
Scanning for LUNs on adapter 0.0.c743
   at port 0x5005076309149145:
      0x4001400300000000
   at port 0x50050763091b9145:
      0x4001400300000000
```

6. If multipathing is not yet configured, complete the following steps:

   a. Install the `device-mapper-multipath`:

```
   # yum -y install device-mapper-multipath
      ...
   Upgraded:
     device-mapper-multipath-0.8.3-3.el8_2.3.s390x
   device-mapper-multipath-libs-0.8.3-3.el8_2.3.s390x
     kpartx-0.8.3-3.el8_2.3.s390x

   Complete!
      ...
```

   b. Copy the multipath reference configuration file to `/etc/multipath.conf`:

```
   # cp -a /usr/share/doc/device-mapper-multipath/multipath.conf
   /etc/multipath.conf
```

   c. Check the status of the `multipathd` daemon. If it is not started, start the service and then, make it permanent by using the following command.

```
   # systemctl status  multipathd
   multipathd.service - Device-Mapper Multipath Device Controller
      Loaded: loaded (/usr/lib/systemd/system/multipathd.service; enabled;
   vendor preset: enabled)
      Active: inactive (dead)
```

```
Condition: start condition failed at Tue 2020-09-29 15:19:56 EDT; 2h 58min
ago
# systemctl start multipathd
# systemctl enable multipathd
```

d. Verify whether multipath set the correct paths to the LUN:

```
# multipath -ll
mpatha (36005076309ffd1450000000000000103) dm-2 IBM,2107900
size=60G features='1 queue_if_no_path' hwhandler='1 alua' wp=rw
`-+- policy='service-time 0' prio=50 status=active
  |- 0:0:0:1073954817 sda 8:0   active ready running
  |- 0:0:1:1073954817 sdb 8:16  active ready running
  |- 1:0:0:1073954817 sde 8:64  active ready running
  `- 1:0:1:1073954817 sdi 8:128 active ready running
```

7. Make the FCP configuration persistent:

```
# lszfcp -D | awk '{ print $1 }' | sed -e 's/\// /g' >> /etc/zfcp.conf
```

8. Check /etc/zfcp.conf file:

```
# cat /etc/zfcp.conf
0.0.b743 0x50050763091b1145 0x4001400300000000
0.0.b743 0x5005076309141145 0x4001400300000000
0.0.c743 0x5005076309149145 0x4001400300000000
0.0.c743 0x50050763091b9145 0x4001400300000000
```

9. Create a partition on the multipath device by using the **parted** command:

```
# parted -s /dev/mapper/mpatha mklabel msdos mkpart primary 0% 100%
```

Note: For more information about **multipathd** on Red Hat Enterprise Linux 8, see *Red Hat Enterprise Linux 8: Configuring device mapper multipath - Using the Device Mapper Multipath feature*.

5.2 Adding a logical volume

Sometimes, you require more disk space than a single direct access storage device (DASD) provides. For example, if you want a shared /home/ directory, it must be of sufficient size for many users to write data to it. When the size is sufficient, you can use the Logical Volume Manager (LVM) to combine multiple DASD volumes into one logical volume. This example does not create a large logical volume, but it shows all the necessary steps.

5.2.1 Creating a logical volume and file system

The following overall steps are involved in creating a logical volume:

1. Create physical volumes from the two partitions.
2. Create a single volume group.
3. Create a single logical volume.
4. Make a file system from the logical volume.

Figure 5-1 shows a block diagram of the LVM.

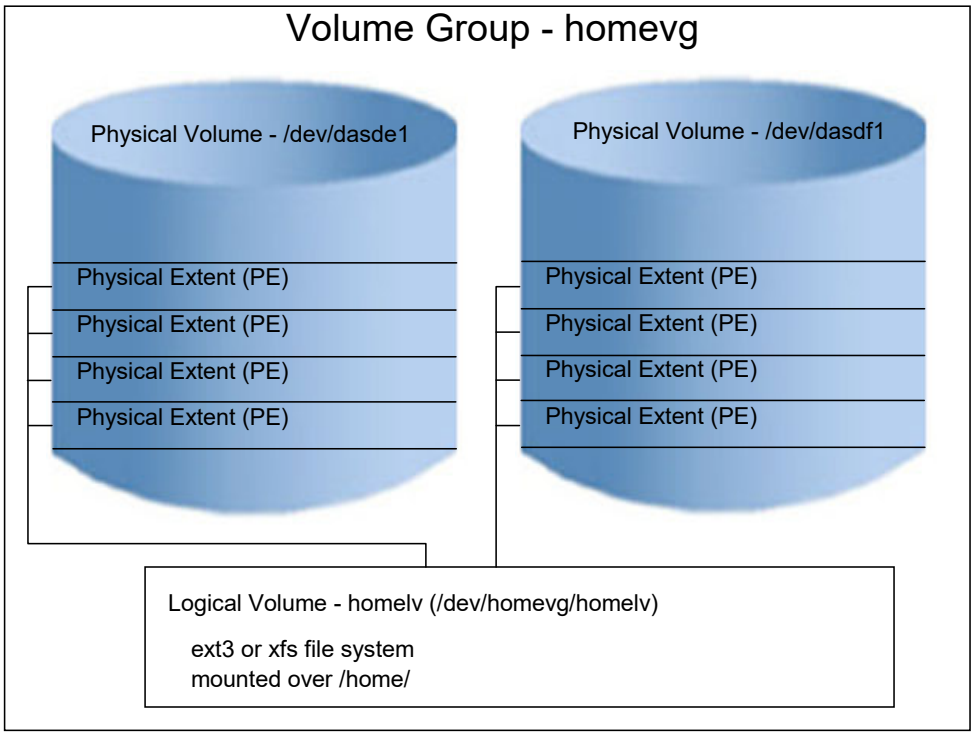

Figure 5-1 LVM block diagram

Creating physical volumes from two minidisks

To create physical volumes from new minidisks at virtual device addresses `103 (dasde)` and `104 (dasdf)`, complete the following steps:

1. Check the devices on your system by using the **`lsdasd`** command.

```
# lsdasd
Bus-ID    Status    Name      Device  Type         BlkSz  Size      Blocks
==============================================================================
0.0.0100  active    dasda     94:0    ECKD         4096   62606MB   16027200
0.0.0300  active    dasdb     94:4    FBA          512    256MB     524288
0.0.0301  active    dasdc     94:8    FBA          512    512MB     1048576
0.0.0102  active    dasdd     94:12   ECKD         4096   7043MB    1803060
0.0.0103  active    dasde     94:16   ECKD         4096   7043MB    1803060
0.0.0104  active    dasdf     94:20   ECKD         4096   7043MB    1803060
```

2. The **`pvcreate`** command initializes partitions for use by LVM. Initialize the two new DASD partitions:

```
# pvcreate /dev/dasde1 /dev/dasdf1
  Physical volume "/dev/dasde1" successfully created.
  Physical volume "/dev/dasdf1" successfully created.
```

3. Verify that the physical volumes were created by using the **`pvdisplay`** command:

```
# pvdisplay /dev/dasde1 /dev/dasdf1
  "/dev/dasde1" is a new physical volume of "<6.88 GiB"
  --- NEW Physical volume ---
  PV Name               /dev/dasde1
  VG Name
  PV Size               <6.88 GiB
```

```
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               rqeQAp-WwKc-W7CD-52zb-P9c1-ZAZa-pld2pJ

  "/dev/dasdf1" is a new physical volume of "<6.88 GiB"
  --- NEW Physical volume ---
  PV Name               /dev/dasdf1
  VG Name
  PV Size               <6.88 GiB
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               JBtZeb-jUYe-7BDU-A65W-1jCx-CcjO-8hibeQ
```

Creating a single volume group

The **vgcreate** command is used to create a volume group that is named homevg from the two partitions. Use the **vgdisplay homevg** command to verify that the volume group was created:

```
# vgcreate homevg /dev/dasde1 /dev/dasdf1
  Volume group "homevg" successfully created

# vgdisplay homevg
  --- Volume group ---
  VG Name               homevg
  System ID
  Format                lvm2
  Metadata Areas        2
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                2
  Act PV                2
  VG Size               13.75 GiB
  PE Size               4.00 MiB
  Total PE              3520
  Alloc PE / Size       0 / 0
  Free  PE / Size       3520 / 13.75 GiB
  VG UUID               TP860g-5ygM-8oEI-rqYm-DtfJ-Xrzr-s7jOM1
```

In this example, 3,520 physical extents (PEs) are free.

Creating a single logical volume

The **lvcreate** command is used to create a logical volume. The **-i** (a lowercase I) specifies the number of stripes. Two stripes are specified in this case because two volumes exist in the volume group. The **-l** (a lowercase L) flag specifies the number of logical extents, which is 3520 in this example. The **-n homelv** flag specifies the name of the new logical volume. The last argument **homevg** specifies the name of the volume group from which the logical volume will be created:

```
# lvcreate -i 2 -l 3520 -n homelv homevg
  Using default stripesize 64.00 KiB.
  Logical volume "homelv" created.
```

Complete the following steps:

Use the **lvdisplay** command to verify that a single logical volume was created. The parameter is the full path of the logical volume, not merely the logical volume name:

```
# lvdisplay /dev/homevg/homelv
  --- Logical volume ---
  LV Path                /dev/homevg/homelv
  LV Name                homelv
  VG Name                homevg
  LV UUID                e08OoU-3DLA-0m4w-ZE1l-cMci-3J25-1LHKWo
  LV Write Access        read/write
  LV Creation host, time rdbkrh04.pbm.ihost.com, 2020-09-21 11:49:20 -0400
  LV Status              available
  # open                 0
  LV Size                13.75 GiB
  Current LE             3520
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     8192
  Block device           253:1
```

Creating a file system from the logical volume

Although various file systems are supported on Red Hat Enterprise Linux 8, XFS is the default and recommended file system.

Create a file system from the new logical volume. The use of xfs is the recommended file system. Create an xfs file system on the new logical volume by using the **mkfs.xfs** command:

```
# mkfs.xfs /dev/homevg/homelv
meta-data=/dev/homevg/homelv     isize=512    agcount=16, agsize=225264 blks
         =                       sectsz=4096  attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1
data     =                       bsize=4096   blocks=3604224, imaxpct=25
         =                       sunit=16     swidth=32 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=4096  sunit=1 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

The file system that was created from the logical volume is now ready to be mounted.

5.2.2 Updating the file system table

You can mount the file system manually. However, if you add the mount to the file system table file, /etc/fstab, you can effectively test the change by using the **mount** command with only one argument. Complete the following steps:

1. Make a backup copy of the file and then add the following line to it:

```
# cd /etc
# cp fstab fstab.works
```

2. Add one line to the fstab file:

```
# vi fstab
... // For RHEL:
/dev/homevg/homelv      /home                   xfs defaults        0 0
```

3. Before you mount the file system over /home/, check that the file system is empty. If a non-root user exists and a new file system is mounted over it, the contents of the directory will be *hidden*. In this example, no data is in the file system:

```
# ls -a /home
.  ..
```

4. Mount the /home/ file system with one argument. By using only one argument, you are testing the change to /etc/fstab. Use the **df -h** command to verify that the file system is mounted (highlighted in bold):

```
# df -k
Filesystem                 1K-blocks    Used Available Use% Mounted on
devtmpfs                     8148104       0   8148104   0% /dev
tmpfs                        8164048       0   8164048   0% /dev/shm
tmpfs                        8164048    8492   8155556   1% /run
tmpfs                        8164048       0   8164048   0% /sys/fs/cgroup
/dev/mapper/vg_system-root  63027132 9328464  53698668  15% /
/dev/dasda1                  1042884  138044    904840  14% /boot
/dev/loop0                   5627134 5627134         0 100% /mnt/rhel82
tmpfs                        1632808       0   1632808   0% /run/user/0
/dev/mapper/homevg-homelv   14406656  134360  14272296   1% /home
```

5. When you add or remove an entry in the /etc/fstab file, run the **systemctl daemon-reload** command for systemd to register the new configuration:

```
# systemctl daemon-reload
```

6. Test a reboot to verify that the new logical volume is successfully mounted over /home/:

```
# reboot
Connection to 129.40.23.135 closed by remote host.
Connection to 129.40.23.135 closed.
```

When the system returns, the new logical volume is mounted over /home/.

5.3 Extending a logical volume

This section describes the process of adding a minidisk to an LVM. This process is useful when your logical volume runs out of space. In this example, we create a large file on the `/var` file system to simulate a file system filling up on RDBKRH04.

Note: The **`fallocate`** utility is used to manipulate the allocated disk space for a file to deallocate or preallocate it.

Use the following commands to create a large file:

```
# df -h /var
Filesystem                  Size  Used Avail Use% Mounted on
/dev/mapper/vg_system-root   61G  8.8G   52G  15% /

# fallocate -l 50G /var/bigfile

# df -h /var
Filesystem                  Size  Used Avail Use% Mounted on
/dev/mapper/vg_system-root   61G   59G  1.4G  98% /
```

A 3390-9 (10017 cylinders) was added as minidisk `104` as described in 5.1, “Adding disk space to virtual machines” on page 96 to accommodate this large file.

Important: You can attach minidisks to a Linux system that is running without rebooting the Linux system. For example, if you added a minidisk at virtual address `104`, from a `root` SSH session, use the **`vmcp link \* 102 102 mr`** command to link to the minidisk. Then, use the **`chccwdev -e 102`** command to enable the minidisk. Do not forget to update /etc/dasd.conf to make it persistent.

To extend the logical volume by using this disk, complete the following steps:

1. Use the **`vgdisplay`** command to see the free space in the volume group `vg_system`:

```
# vgdisplay vg_system
  --- Volume group ---
  VG Name               vg_system
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  2
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               1
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               <60.14 GiB
  PE Size               4.00 MiB
  Total PE              15395
  Alloc PE / Size       15395 / <60.14 GiB
  Free  PE / Size       0 / 0
  VG UUID               htUmTy-nLxL-ZibJ-U9wL-yWkZ-tObq-mmyL1U
```

No free extents are in the volume group.

2. Use the **lsdasd** command to show the enabled disks:

```
# lsdasd
Bus-ID    Status    Name      Device  Type         BlkSz  Size      Blocks
==============================================================================
=
0.0.0100  active    dasda     94:0    ECKD         4096   62606MB   16027200
0.0.0103  active    dasdb     94:4    ECKD         4096   7043MB    1803060
0.0.0301  active    dasdc     94:8    FBA          512    512MB     1048576
0.0.0102  active    dasdd     94:12   ECKD         4096   7043MB    1803060
0.0.0300  active    dasde     94:16   FBA          512    256MB     524288
0.0.0104  active    dasdf     94:20   ECKD         4096   7043MB    1803060
```

Minidisk 102 is at /dev/dasdd.

3. Make minidisk 102 a physical volume by using the **pvcreate** command:

```
# pvcreate /dev/dasdd1
   Physical volume "/dev/dasdd1" successfully created.
```

4. Use the **vgextend** command to add the minidisk to the volume group:

```
# vgextend vg_system /dev/dasdd1
   Volume group "vg_system" successfully extended
```

5. Use the **vgdisplay** command again to show the free extents in the volume group:

```
# vgdisplay vg_system
  --- Volume group ---
  VG Name               vg_system
  System ID
  Format                lvm2
  Metadata Areas        2
  Metadata Sequence No  9
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               1
  Max PV                0
  Cur PV                2
  Act PV                2
  VG Size               67.01 GiB
  PE Size               4.00 MiB
  Total PE              17155
  Alloc PE / Size       15395 / <60.14 GiB
  Free  PE / Size       1760 / <6.88 GiB
  VG UUID               htUmTy-nLxL-ZibJ-U9wL-yWkZ-tObq-mmyL1U
```

Now, 1,760 free extents are in the volume group.

6. Use the **df** command to determine the name of the logical volume that is used to hold /var directory:

```
# df -h /var/
Filesystem                  Size  Used Avail Use% Mounted on
/dev/mapper/vg_system-root   61G   59G  1.4G  98% /
```

In this example, the /var does not have its own logical volume. However, the output that is shown in the /var folder is under /dev/mapper/vg_system-root logical volume. Therefore, we perform the steps to extend this file system.

7. Use the **lvextend** command to extend the volume group with all of the new extents:

```
# lvextend -l +1760 /dev/mapper/vg_system-root
  Size of logical volume vg_system/root changed from <60.14 GiB (15395 extents)
to 67.01 GiB (17155 extents).
  Logical volume vg_system/root successfully resized.
```

8. Use the **fsadm resize** command to increase the size of the XFS file system while it is still mounted:

```
# fsadm resize /dev/mapper/vg_system-root
meta-data=/dev/mapper/vg_system-root isize=512    agcount=4, agsize=3941120
blks
         =                       sectsz=4096  attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1
data     =                       bsize=4096   blocks=15764480, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=7697, version=2
         =                       sectsz=4096  sunit=1 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
data blocks changed from 15764480 to 17566720
```

Note: You also can use the **xfs_growfs** command to increase the size of the XFS file system while it is still mounted.

9. Use the **df** command to show the file system size before and after you extend it, as shown in the following example:

```
# df -h /var
Filesystem                  Size  Used Avail Use% Mounted on
/dev/mapper/vg_system-root   67G   59G  8.2G  88% /
```

The file system (/dev/mapper/vg_system-root) where /var/ is located now has 8.2 GB of free space. The root file system was extended by using a new disk. If you reboot the Linux server now, your system does not come up because the 102 disk is not added into the **initramfs** image.

In this case, your server enters emergency mode, as shown in Example 5-7.

Example 5-7 Entering emergency mode

```
[  219.070488] dracut-initqueue[450]: Warning: Could not boot.
         Starting Setup Virtual Console...
[ [0;32m  OK   [0m] Started Setup Virtual Console.
         Starting Dracut Emergency Shell...
Warning: /dev/mapper/vg_system-root does not exist
Warning: /dev/vg_system/root does not exist

Generating "/run/initramfs/rdsosreport.txt"

Entering emergency mode. Exit the shell to continue.
Type "journalctl" to view system logs.
You might want to save "/run/initramfs/rdsosreport.txt" to a USB stick or /boot
after mounting them and attach it to a bug report.
```

Rebuild the **initramfs** image by using the **dracut -f** command and then, issue **zipl** to prepare the boot device:

```
# dracut -f

# zipl
Using config file '/etc/zipl.conf'
Using BLS config file
'/boot/loader/entries/95742106573a4215ac86da4a36c81e40-4.18.0-193.el8.s390x.con
f'
Using BLS config file
'/boot/loader/entries/95742106573a4215ac86da4a36c81e40-0-rescue.conf'
Building bootmap in '/boot'
Building menu 'zipl-automatic-menu'
Adding #1: IPL section 'Red Hat Enterprise Linux (4.18.0-193.el8.s390x) 8.2
(Ootpa)' (default)
Adding #2: IPL section 'Red Hat Enterprise Linux
(0-rescue-95742106573a4215ac86da4a36c81e40) 8.2 (Ootpa)'
Preparing boot device: dasda (0100).
Done.
```

Tip: If you forget to issue the **dracut -f** and **zipl** commands to update the boot partition, it is likely your server enters emergency mode when rebooted. In this case, issue the following commands in the emergency console to bring the 102 disk online and recover the server:

```
# cio_ignore -r 102
# echo 1 > /sys/bus/ccw/drivers/dasd-eckd/0.0.0102/online
# lvm vgchange -ay
# systemctl default
```

After the server is up, proceed with **dracut -f** and **zipl** commands.

For more information about recovering a Red Hat Enterprise Linux server, see 11.1, “Rescuing a Linux system” on page 202.

5.4 Moving a physical volume

In addition to file systems growing, you might need to move data off one or more volumes onto another volume or a target set of volumes. If your data is in LVM, the **pvmove** and **vgreduce** commands were designed for this process. They can be used with the file system online.

This procedure is helpful during storage migrations and you can perform this activity while applications are up and running. Consider the use of this approach If your application is critical and you do not want to perform an offline storage migration.

In this example, we use the **homevg** volume group that was created as described in 5.2.1, “Creating a logical volume and file system” on page 103 to reproduce the necessary steps to move data from one physical volume to another. This process is performed while the file system is online.

Complete the following steps:

1. Add a DASD minidisk as 105. If necessary, follow instructions in 5.1, “Adding disk space to virtual machines” on page 96.
2. List all DASD disks by using the following command:

```
# lsdasd
Bus-ID    Status    Name      Device  Type         BlkSz  Size      Blocks
==============================================================================
0.0.0100  active    dasda     94:0    ECKD         4096   62606MB   16027200
0.0.0103  active    dasdb     94:4    ECKD         4096   7043MB    1803060
0.0.0301  active    dasdc     94:8    FBA          512    512MB     1048576
0.0.0102  active    dasdd     94:12   ECKD         4096   7043MB    1803060
0.0.0300  active    dasde     94:16   FBA          512    256MB     524288
0.0.0104  active    dasdf     94:20   ECKD         4096   7043MB    1803060
0.0.0105  active    dasdg     94:24   ECKD         4096   7043MB    1803060
```

3. Add the 105 minidisk into the LVM:

```
# pvcreate /dev/dasdg1
  Physical volume "/dev/dasdg1" successfully created.
```

4. Display the number of current physical extents:

```
# vgdisplay homevg | grep "Total PE"
  Total PE              3520
```

5. Display the available space in /home:

```
# df -h /home
Filesystem                Size  Used Avail Use% Mounted on
/dev/mapper/homevg-homelv   14G  132M   14G   1% /home
```

6. Create a sizable file on it by using the **dd** command and show the file system usage:

```
# fallocate -l 12G /home/bigfile

# df -h /home
Filesystem                Size  Used Avail Use% Mounted on
/dev/mapper/homevg-homelv   14G   13G  1.7G  89% /home
```

7. Show the volume group usage by using the **vgdisplay** command:

```
# vgdisplay homevg
  --- Volume group ---
  VG Name               homevg
  System ID
  Format                lvm2
  Metadata Areas        2
  Metadata Sequence No  2
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               1
  Max PV                0
  Cur PV                2
  Act PV                2
  VG Size               13.75 GiB
  PE Size               4.00 MiB
  Total PE              3520
  Alloc PE / Size       3520 / 13.75 GiB
```

```
  Free  PE / Size       0 / 0
  VG UUID               TP860g-5ygM-8oEI-rqYm-DtfJ-Xrzr-s7jOM1
```

All of the physical extents in the volume group are used.

8. Add a third physical volume, which is the target of the data move, to the volume group:

```
# vgextend homevg /dev/dasdg1
  Volume group "homevg" successfully extended
```

9. Show the volume group usage again:

```
# vgdisplay homevg
  --- Volume group ---
  VG Name               homevg
  System ID
  Format                lvm2
  Metadata Areas        3
  Metadata Sequence No  3
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               1
  Max PV                0
  Cur PV                3
  Act PV                3
  VG Size               20.62 GiB
  PE Size               4.00 MiB
  Total PE              5280
  Alloc PE / Size       3520 / 13.75 GiB
  Free  PE / Size       1760 / <6.88 GiB
  VG UUID               TP860g-5ygM-8oEI-rqYm-DtfJ-Xrzr-s7jOM1
```

The volume group was increased to approximately 6.88 GB.

10. Use the **pvscan** command to confirm the target disk has available space to receive the data before the **pvmove** command is used:

```
# pvscan
  PV /dev/dasdf1   VG homevg          lvm2 [<6.88 GiB / 0    free]
  PV /dev/dasdb1   VG homevg          lvm2 [<6.88 GiB / 0    free]
  PV /dev/dasdg1   VG homevg          lvm2 [<6.88 GiB / <6.88 GiB free]
  PV /dev/dasda2   VG vg_system       lvm2 [<60.14 GiB / 0    free]
  PV /dev/dasdd1   VG vg_system       lvm2 [<6.88 GiB / 0    free]
  Total: 5 [<87.64 GiB] / in use: 5 [<87.64 GiB] / in no VG: 0 [0   ]
```

11. Move the data off the /dev/dasdf1 physical volume by using the **pvmove** command (the target does not need to be specified):

```
# pvmove /dev/dasdf1
  /dev/dasdf1: Moved: 0.00%
  /dev/dasdf1: Moved: 32.78%
  /dev/dasdf1: Moved: 65.51%
  /dev/dasdf1: Moved: 98.30%
  /dev/dasdf1: Moved: 100.00%
```

12. Show the volume group usage again:

```
# vgdisplay homevg
  --- Volume group ---
  VG Name               homevg
  System ID
  Format                lvm2
  Metadata Areas        3
  Metadata Sequence No  6
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               1
  Max PV                0
  Cur PV                3
  Act PV                3
  VG Size               20.62 GiB
  PE Size               4.00 MiB
  Total PE              5280
  Alloc PE / Size       3520 / 13.75 GiB
  Free  PE / Size       1760 / <6.88 GiB
  VG UUID               TP860g-5ygM-8oEI-rqYm-DtfJ-Xrzr-s7jOM1
```

The free and used extents are the same; however, the data was moved.

13. Show the free and used extents on the source and target physical volumes by using the **pvdisplay** command:

```
# pvdisplay /dev/dasdf1 /dev/dasdg1
  --- Physical volume ---
  PV Name               /dev/dasdf1
  VG Name               homevg
  PV Size               <6.88 GiB / not usable <3.11 MiB
  Allocatable           yes
  PE Size               4.00 MiB
  Total PE              1760
  Free PE               1760
  Allocated PE          0
  PV UUID               rqeQAp-WwKc-W7CD-52zb-P9c1-ZAZa-pld2pJ

  --- Physical volume ---
  PV Name               /dev/dasdg1
  VG Name               homevg
  PV Size               <6.88 GiB / not usable <3.11 MiB
  Allocatable           yes (but full)
  PE Size               4.00 MiB
  Total PE              1760
  Free PE               0
  Allocated PE          1760
  PV UUID               9JQYjX-2OFy-ld2T-mWNs-Zd81-fNcl-pOvLUD
```

14. Remove the source physical volume:

```
# vgreduce homevg /dev/dasdf1
  Removed "/dev/dasdf1" from volume group "homevg"
```

The source volume is now ready for reassignment, or retirement.

6

Monitoring z/VM and Linux

This chapter briefly describes how to monitor z/VM and Linux.

For more information about z/VM performance and monitoring, see Chapter 11, “Monitoring performance and capacity” in *Getting Started with Linux on IBM Z*, SC24-6287.

Many z/VM monitoring tools, such as CA VM, Monitor, IBM z/VM Performance Toolkit, IBM Tivoli® OMEGAMON® XE for z/VM and Linux, and products from IBM Velocity Software are available. The IBM z/VM Performance Toolkit is briefly described in this chapter.

This chapter includes the following topics:

6.1 Using basic z/VM commands

z/VM features many commands that are used to monitor the state of the system. The **CP INDICATE** command is the most commonly used command. Other commands are also addressed in this book. For more information, see the z/VM Performance Resource web page.

6.1.1 INDICATE command

In z/VM, use the **INDICATE** command. Many **INDICATE** command parameters can be included as command-line options. Use the **HELP INDICATE** command for a basic understanding and then press F11 for help with each parameter.

INIDICATE LOAD

If no parameter is specified, the **INDICATE LOAD** command is the default option. Two versions exist, depending on whether the issuing virtual machine has privilege class G or class E. Class G users can use the **INDICATE** command to display recent contention for system resources, display environment characteristics, and measurements of resources that are used by their virtual machine.

The output from virtual machines with class E privilege (for example, `MAINT` or `OPERATOR`) is shown (the lines are numbered for clarity with a description of the output):

```
===> ind load
1  AVGPROC-000% 04
2  MDC READS-000068/SEC WRITES-000001/SEC HIT RATIO-099%
3  PAGING-0/SE
4  Q0-00001(00000)                          DORMANT-00012
5  Q1-00000(00000)           E1-00000(00000)
6  Q2-00001(00000) EXPAN-001 E2-00000(00000)
7  Q3-00001(00000) EXPAN-001 E3-00000(00000)
8
9  PROC 0000-000% CP   VM     PROC 0001-000% CP   VL
10 PROC 0002-000% IFL  VM     PROC 0003-000% IFL  VL
11
12 LIMITED-00000
```

The **INDICATE LOAD** command gives a snapshot of the current system performance. Except for the counts of virtual machines in various queues and the limited list, the values that you see are a smoothed average over the past 4 minutes. z/VM performance analysts tend to focus on the following areas:

- `AVGPROC` on line **1** gives the overall processor utilization, which is `38%` in this example. The number that follows the utilization is the number of online processors, which is `04` in this example.

  The individual processor utilization is shown on lines **9** and **10**. Review these lines to see whether they are balanced. Sometimes, an imbalance is acceptable, for example, low utilization scenarios or cases where insufficient users are ready to run virtual processors to keep the physical processors busy.

  One of the processors is a Master. All of the other processors are Alternate. An imbalance can result from performing these functions. Another imbalance can result from vertical CPU management.

- (MDC) statistics are on the second line. The effectiveness of MDC can be judged by the combination of the READS rate and the HIT RATIO. If both are high, many physical I/Os are avoided because of the MDC feature.

  The real, physical I/O avoidance can be high for a system that has an high I/O rate, which is composed of reads plus writes, a high proportion of reads, and a good hit ratio for those reads (tending to 90% or greater). Cases of avoidance as high as 50% were documented. However, a high HIT RATIO with a low value for the READS rate is not good (100% hit ratio when you perform only 1 I/O per second [IOPS] is effectively meaningless).
- Line **3** describes more storage (memory) management. The PAGING rate is important. Higher values often affect performance. This effect can be at least partially offset by increasing the number of page volumes, but a more thorough examination of this problem is advisable whenever it arises.
- On lines **4** - **7**, you also see a series of counters that represent the users in various queues. The z/VM scheduler classifies work into three classes (1 - 3) and a special extra class that is labeled zero. Therefore, the column of Q_x values and E_x represents the virtual machines in the dispatch list and the eligible list.

  The most important value to validate is that no virtual machines are in the Eligible list: E1, E2, and E3. This condition implies that z/VM stopped dispatching virtual machines to avoid overcommitted resources. This system requires further investigation, which possibly leads to tuning work, or even more hardware in extreme cases. Do not be concerned about the values in parentheses.

INDICATE QUEUES EXP

Another useful command to understand the state of the system is the **INDICATE QUEUES EXP** command, for example:

```
===> ind q exp
MAINT         Q1 R00   00001623/00001552   .I..  .0004
TCPIP         Q0 PS    00003496/00003178   .I..  99999
```

This command is another class E command. It displays the virtual processors that are associated with a specific virtual machine (that can have multiple virtual processors), the queue (dispatch list, eligible list, or limit list) that they are in, and the state that they are in. This output is a snapshot in time.

Again, you want to check this output to ensure that no virtual machines are in the eligible list. Normal virtual processors in the dispatch list are Q_x (x = 1, 2, or 3). The eligible list is marked as E_x.

The third column in the example also provides the state of the virtual processor, which is helpful to identify how the virtual processors might be constrained. Virtual processors that are running during the snapshot period are marked with an RNN where NN is the processor number they are on. An R without a number means that the virtual processor is ready to run, but no processor is available.

Note: The virtual machine that issues the **INDICATE** command is always one of the running machines.

Other states are documented in the help for the **IND Q EXP** command. Do not worry about the other columns unless detailed analysis is required or if IBM support requests the information.

Also, this output is only a snapshot in time so repeat this command often over time for a more accurate picture of your z/VM system. A single snapshot cannot be regarded as indicative.

6.1.2 Other basic commands

Other basic commands are also useful. All of the examples that are shown in this section are run from the MAINT virtual machine. The results differ for users with fewer privileges.

Getting help

To obtain help on the system, use the **HELP** command. Sometimes, it is hard to identify help for the exact command that you are looking for. The following **help** commands are useful:

```
===> help             // for basic help
===> help menus       // for a menu of all z/VM help menus
===> help cp menu     // for a menu of all CP commands
===> help cpquery     // for a menu of all CP QUERY commands
===> help cpset       // for a menu of all CP SET commands
```

Determining who is logged on

To see who is logged on to the system, use the **QUERY NAMES** command, for example:

```
===> q n
DIRMSAT2 - SSI
ZMAPVM62 - DSC , LINUX153 - DSC , LNXADMIN - DSC , LINUX157 - DSC
VSMEVSRV - DSC , VSMPROXY - DSC , VSMREQIU - DSC , VSMREQI6 - DSC
VSMREQIN - DSC , DTCSMAPI - DSC , PERSMAPI - DSC , VSMWORK3 - DSC
VSMWORK2 - DSC , VSMWORK1 - DSC , FTPSERVE - DSC , VSMGUARD - DSC
TCPIP    - DSC , DIRMAINT - DSC , DTCVSW2  - DSC , DTCVSW1  - DSC
VMSERVP  - DSC , VMSERVR  - DSC , VMSERVU  - DSC , VMSERVS  - DSC
OPERSYMP - DSC , DISKACNT - DSC , EREP     - DSC , OPERATOR - DSC
MAINT    -L0004
VSM     - TCPIP
```

Determining storage or memory

To see how much main storage (memory) is installed and allocated to a system, use the **QUERY STORAGE** command, for example:

```
===> q stor
STORAGE = 16G CONFIGURED = 16G INC = 256M STANDBY = 0  RESERVED = 0
```

This command output shows 16 GB of central memory (storage).

Determining processors or central processor units

To see how many processors (central processor units [CPUs] or Integrated Facilities for Linux [IFLs]) are allocated at the system level, use the **QUERY PROCESSORS** command, for example:

```
===> q proc
PROCESSOR 00 MASTER CP
PROCESSOR 01 ALTERNATE CP
PROCESSOR 02 ALTERNATE CP
PROCESSOR 03 ALTERNATE CP
PROCESSOR 04 ALTERNATE CP
PROCESSOR 05 ALTERNATE CP
PROCESSOR 06 ALTERNATE CP
PROCESSOR 07 ALTERNATE CP
PROCESSOR 08 ALTERNATE CP
PROCESSOR 09 ALTERNATE CP
```

Determining software level

To determine the control program (CP) level for your system, use the **QUERY CPLEVEL** command, for example:

```
===> q cplevel
z/VM Version 6 Release 3.0, service level 1301 (64-bit)
Generated at 06/28/13 14:58:28 EDT
IPL at 09/04/13 10:48:34 EDT
```

Determining system cylinder allocation

The **QUERY ALLOC MAP** command shows you the system allocation of spool, paging, and directory space, for example:

```
===> q alloc map
               EXTENT     EXTENT                        % ALLOCATION
VOLID  RDEV     START        END  TOTAL IN USE   HIGH USED TYPE
------ ---- ---------- ---------- ------ ------ ------ ---- -------------
JV1030 1030          1         20     20      1      1   5% DRCT ACTIVE
JV1031 1031          1       3338 600840  87022  91029  14% SPOOL
JV1131 1131          -          -      0      0      0   0  SHARED
JP1260 1260          0      10016  1761K     27     56   1% PAGE
JP1261 1261          0      10016  1761K     75     75   1% PAGE
JV1032 1032          1       3338 600840     52     63   1% PAGE
```

Determining DASD, Open Systems Adapter, and virtual resources

The **QUERY DASD** and **QUERY DASD FREE** commands show you the DASD that is assigned to the system and the DASD that is free to be assigned. Similarly, the **QUERY OSA** and **QUERY OSA FREE** commands report on the Open Systems Adapter (OSA) resources. The **QUERY VIRTUAL ALL** command is useful. The following list shows the short form of these commands without any output:

```
===> q da
===> q da free
===> q osa
===> q osa free
===> q v all
```

6.2 IBM z/VM Performance Toolkit

The z/VM Performance Toolkit is a part of the z/VM base installation and is installed as disabled. Because this tool kit is priced feature of z/VM, you must order the z/VM Performance Toolkit, which must be configured

For information about z/VM Performance Toolkit, see the following publications:

- *z/VM Performance Toolkit Guide*, SC24-6302, and *z/VM Performance Toolkit Reference*, SC24-6303.
- *The Program Directory for Performance Toolkit for VM*, GI10-0785
- *Linux on IBM z Series and S/390: Performance Toolkit for VM*, SG24-6059

6.2.1 Configuring Performance Toolkit for VM

Performance Toolkit is installed with z/VM. The configuration is described in the Program Directory for Performance Toolkit for VM.

Complete the following steps to activate Performance Toolkit:

1. Query the priced products that are enabled with the **QUERY PRODUCT** command:

```
===> q product
Product  State    Description
IBMVMSSI Enabled  IBM z/VM Single System Image Feature
6VMDIR30 Disabled 00/00/00.00:00:00.$BASEDDR DIRECTORY MAINTENANCE FACILITY
(Dir
Maint)
6VMPTK30 Disabled 00/00/00.00:00:00.$BASEDDR PERFORMANCE TOOLKIT FOR VM
6VMRAC30 Disabled 00/00/00.00:00:00.$BASEDDR RACF Security Server
6VMRSC30 Disabled 00/00/00.00:00:00.$BASEDDR RSCS Networking
```

2. To enable IBM Performance Toolkit for VM, log on to **MAINT630** and enter the following command:

```
===> service perftk enable
VMFSRV2760I SERVICE processing started
...
VMFSUT2760I VMFSUFTB processing started
VMFSUT2760I VMFSUFTB processing completed successfully
VMFSRV2760I SERVICE processing completed successfully
```

   A few windows of messages scroll by. Finally, the success messages are shown. Performance Toolkit is enabled for the current z/VM session.

3. The `SYSTEM CONFIG` file is modified by appending a line to the end of the file. Verify that this line was added with the following commands:

```
===> vmlink pmaint cf0
DMSVML2060I PMAINT CF0 linked as 0120 file mode Z

===> type system config z
...    // many screens cleared
PRODUCT PRODID 6VMPTK30 STATE ENABLED DESCRIPTION '06/05/13.15:22:55.MAINT630
PE
RFKIT Minidisk Install and Service'
```

   The **QUERY PRODUCT** command shows the change:

```
===> q product
Product  State    Description
IBMVMSSI Enabled  IBM z/VM Single System Image Feature
6VMDIR30 Disabled 00/00/00.00:00:00.$BASEDDR DIRECTORY MAINTENANCE FACILITY
(Dir
Maint)
6VMPTK30 Enabled  06/05/13.15:22:55.MAINT630 PERFKIT Minidisk Install and
Service
6VMRAC30 Disabled 00/00/00.00:00:00.$BASEDDR RACF Security Server
6VMRSC30 Disabled 00/00/00.00:00:00.$BASEDDR RSCS Networking
```

The Performance Toolkit is now enabled. You also can verify that the Performance Toolkit is enabled by running the **QUERY PRODUCT** command again.

6.2.2 Configuring web browser support

After the product is enabled, modify the TCP/IP profile to enable web access to the Performance Toolkit. The following example sets the port to 80, which is the default for a web browser:

1. Log on to TCPMAINT.
2. Edit the TCPIP configuration file. The default name is PROFILE TCPIP. Search for the string reserve ports, which is where the z/VM TCP/IP ports are reserved:

```
===> x profile tcpip d
====> /port
```

3. Add the following line under the PORT entries:

```
...
PORT
  20   TCP FTPSERVE  NOAUTOLOG ; FTP Server
  21   TCP FTPSERVE            ; FTP Server
  23   TCP INTCLIEN            ; TELNET Server
; 25   TCP SMTP                ; SMTP Server
  80   TCP PERFSVM             ; Performance Toolkit
; 111  TCP PORTMAP             ; Portmap Server
; 111  UDP PORTMAP             ; Portmap Server
; 143  TCP IMAP                ; IMAP Server
...
```

4. Save your changes.
5. To change TCP/IP dynamically, use the **OBEYFILE** command:

```
===> netstat obey port 80 tcp perfsvm
VM TCP/IP Netstat Level 630       TCP/IP Server Name: TCPIP

OBEY command response is: OK
OBEY return code = 0
```

6. Issue the **NETSTAT CLIENTS** command to verify your configuration. Ensure that the service that is named PERFSVM is a client. Client: PERFSVM shows after a few windows of output:

```
===> netstat clients
...
Client: PERFSVM                 Authorization: {none}
Notes Handled: none
Last Touched:   0:03:23
Vmcf error count: 0
```

If you are configuring central monitoring in a single system image (SSI) cluster, configuring the web server on only one of the members is sufficient. Central monitoring enables one member to monitor the other members of the SSI cluster.

6.2.3 Configuring PERFSVM

The PERFSVM virtual machine is the Performance Toolkit service machine. Complete the following steps to configure it:

1. Log on to PERFSVM. If you successfully enabled the product, you enter a Performance Toolkit session and see the following text at the top of the window:

```
FCX001                Performance Toolkit for VM               Autoscroll 12
 FCXBAS500I Performance Toolkit for VM FL630
 16:14:15 Monitor event started -- recording is activated
 16:14:15 Monitor sample started -- recording is activated
```

2. Press F12 twice to reach a Conversational Monitor System (CMS) prompt.

3. Copy the default configuration files, which are on the PERFSVM D disk, to your A disk:

```
===> copy * * d = = a
```

4. The main configuration file is FCONX $PROFILE. Edit that file and search for the following string VMCF:

```
===> x fconx $profile
====> /vmcf
```

   This search takes you to line 190 where the next eight lines are comments that start with an asterisk (*). Make the following changes:

   – Uncomment the second, fourth, sixth, and eighth lines by changing *C to **FC**.
   – Change port 81 to **80** on the fourth line so that you can use a browser interface without the need to specify port 81 on the URL (with a :81 suffix).

   The modified lines then resemble the following lines. Save your changes by using the **FILE** subcommand:

```
*    Following command activates VMCF data retrieval interface
FC MONCOLL VMCF ON
*    Define the maximum allowed number of Internet connections
FC MONCOLL WEBSERV MAXCONN 100
*    Define the timeout of inactive Internet connections in minutes
FC MONCOLL WEBSERV TIMEOUT 30
*    Following command activates Internet interface
FC MONCOLL WEBSERV ON TCPIP TCPIP 80
*    Following command activates Internet interface with SSL
...
====> file
```

   If you are configuring central monitoring in an SSI cluster, enable the four FC commands on one member only, which serves as a web server. On the other members, allow only the first FC statement (**FC MONCOLL VMCF ON**).

5. Create a remote data retrieval authorization file with your z/VM system identifier (replace ***ZVM63A*** with your system identifier):

```
===> x fconrmt authoriz
====> a 2
ZVM63A PERFSVM  S&FSERV DATA
```

   If you are configuring central monitoring in an SSI cluster, allow the member that serves as the web server to access the other members. The authorization file on a second member resembles the following example:

```
ZVM63A PERFSVM  DATA
ZVM63B PERFSVM  S&FSERV DATA
```

6. Create a system identification file that links your z/VM systems and PERFSVM to a special resource name (replace ***ZVM63A*** with your system identifier):

```
===> x fconrmt systems
====> a
ZVM63A PERFSVM  z/VM6.3 N FCXC1RO1
```

If you are configuring central monitoring in an SSI cluster, also specify all other members. Ensure that each member uses a unique resource name. The first member can be **FCXC1RO1**, the second member can be **FCXC1RO2**, and so on:

```
ZVM63A PERFSVM  z/VM6.3 N FCXC1RO1
ZVM63B PERFSVM  z/VM6.3 N FCXC1RO2
ZVM63C PERFSVM  z/VM6.3 N FCXC1RO3
ZVM63D PERFSVM  z/VM6.3 N FCXC1RO4
```

Note: The system identification files on all members must be the same.

7. Set up a resource override for the default resource name (enter the resource name that you used in FCONRMT AUTHORIZ):

```
===> x ucomdir names
====> a 6
:nick.FCXRESOO :luname.*IDENT
            :tpn.FCXC1RO1
            :security.SAME
:nick.FCXSYSTM :luname.*IDENT
            :tpn.FCXC1SO1
            :security.SAME
```

If you are configuring central monitoring in an SSI cluster, specify resource override on each member. The second member uses **FCXC1RO2** and **FCXC1SO2**. The third member uses **FCXC1RO3** and **FCXC1SO3**. The fourth member uses **FCXC1RO4** and **FCXC1SO4**.

8. Start CP to collect performance data and start Performance Toolkit automatically after the IPL:

   a. Log on to AUTOLOG1.

   b. Before you press Enter at the VM READ prompt, enter acc (noprof so that the **PROFILE EXEC** is not run:

```
LOGON AUTOLOG1
z/VM Version 6 Release 3.0, Service Level 0000 (64-bit),
built on IBM Virtualization Technology
There is no logmsg data
FILES:   NO RDR, 0008 PRT,   NO PUN
LOGON AT 12:13:55 EDT THURSDAY 06/06/13
z/VM V6.3.0    2013-06-04 12:50
acc (noprof
Ready; T=0.01/0.01 12:14:01
```

   c. Edit the profile exec as shown in the following example:

```
===> x profile exec a
...
/*****************************************************************/
/* Customer processing can be added here                         */
/*****************************************************************/
"CP XAUTOLOG TCPIP"              /* Autolog TCPIP                */
"CP SET MDC STOR 0M 256M"        /* Limit minidisk cache in CSTOR*/
```

```
"CP SET SIGNAL SHUTDOWN 600"     /* Allow guests 10 min to shut down */
"CP XAUTOLOG LNXADMIN"           /* Start the Linux admin machine    */

"CP MONITOR SAMPLE ENABLE PROCESSOR" /* Setup CP MONITOR parameters  */
"CP MONITOR SAMPLE ENABLE STORAGE"
"CP MONITOR SAMPLE ENABLE USER ALL"
"CP MONITOR SAMPLE ENABLE I/O ALL"
"CP MONITOR SAMPLE ENABLE NETWORK"
"CP MONITOR SAMPLE ENABLE APPLDATA ALL"
"CP MONITOR SAMPLE ENABLE ISFC"
"CP MONITOR SAMPLE ENABLE SSI"

"CP MONITOR EVENT  ENABLE STORAGE"
"CP MONITOR EVENT  ENABLE I/O ALL"
"CP MONITOR EVENT  ENABLE NETWORK"
"CP MONITOR EVENT  ENABLE ISFC"
"CP MONITOR EVENT  ENABLE SSI"

"CP MONITOR SAMPLE INTERVAL 1 MIN"  /* Set sampling interval */

"CP XAUTOLOG PERFSVM"            /* Start Performance Toolkit        */
```

d. Save the file by using the following command:

```
====> file
```

> **Note:** If you do not plan to IPL before you try Performance Toolkit, run all **CP MONITOR** commands that you added to the **PROFILE EXEC** file so that the CP starts to collect performance data.

e. Log off from AUTOLOG1.

6.2.4 Starting IBM Performance Toolkit for VM

To start Performance Toolkit, complete the following steps:

1. Log on to the PERFSVM virtual machine.
2. Press Enter and the Performance Toolkit starts through the PROFILE EXEC:

```
FFCX001                Performance Toolkit for VM               Autoscroll 12
 FCXBAS500I Performance Toolkit for VM FL630
 12:32:15 FCXAPP530I Connected to *IDENT for resource FCXC1R01
 12:32:15 FCXAPF530I Connected to *IDENT for resource FCXC1S01
 12:32:15 FCXTCP571I Connected to TCP/IP server TCPIP on path 0003
 12:32:15 FCXAPP527I User PERFSVM connected on path 0006
 12:32:15 FCXAPC535I Connected to resource FCXC1R01 on path 0005, for S&F-Coll
 12:32:15 FCXTCP575I WebServer host IP address is 9.12.7.11:00080
 12:32:15 FCXTCP590I WebServer interface activated
 12:32:15 Monitor event started -- recording is activated
 12:32:15 Monitor sample started -- recording is activated
```

3. Disconnect from PERFSVM now:

```
Command ===> disc
```

Performance Toolkit is now configured and running.

6.2.5 Using IBM Performance Toolkit for VM

Performance Toolkit can be used with a web browser or 3270 interface.

Using a web browser interface

Complete the following steps to use web-enabled Performance Toolkit:

1. Point a browser to your z/VM system, for example:

   `http://9.12.7.11`

   You see a splash window, and then, the Web Server log-on window, as shown in Figure 6-1.

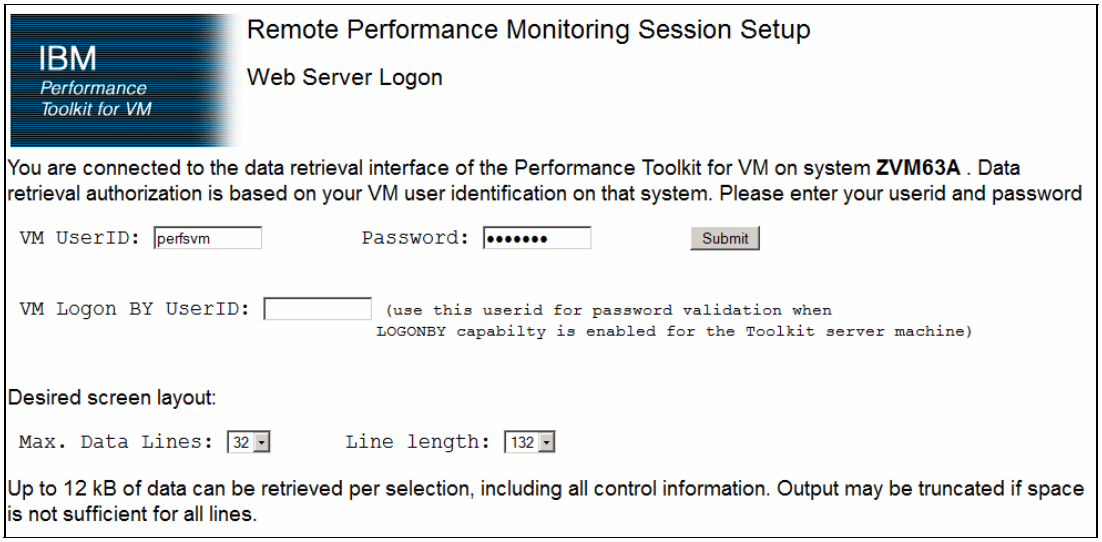

Figure 6-1 Performance Toolkit log-on window

2. Enter any valid user ID and password and click **Submit**. In this example, we use `PERFSVM`.

   The Central Monitoring System Load Overview window opens with your system identifiers (*Node-ID*) on the left side.

3. Click your system identifier and the Initial Performance Data Selection Menu window opens, as shown in Figure 6-2 on page 126.

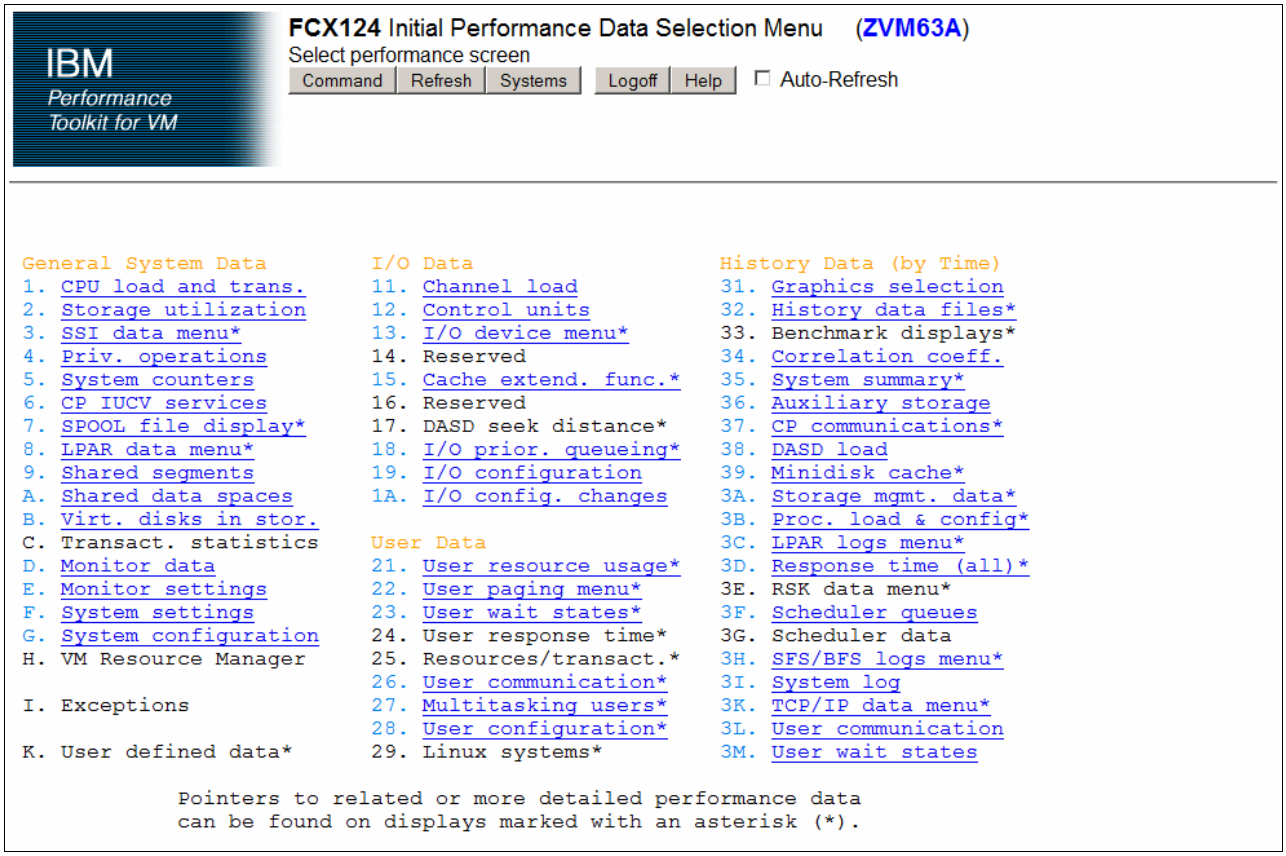

Figure 6-2 Browser interface to Performance Toolkit

From this window, you can drill down into many types of reports.

Using a 3270 interface

Complete the following steps to use a 3270 interface:

1. Log on to `PERFSVM`.
2. If you disconnected, press Enter to return to the Performance Toolkit command line. If the virtual machine was logged off, the **`PROFILE EXEC`** runs and returns you to the command line. Enter the **`MONITOR`** command:

   ```
   Command ===> monitor
   ```

   The Performance Screen Selection panel opens, as shown in Example 6-1.

Example 6-1 Performance Screen Selection panel

```
FCX124          Performance Screen Selection  (FL630        )    Perf. Monitor

General System Data        I/O Data                  History Data (by Time)
1. CPU load and trans.     11. Channel load          31. Graphics selection
2. Storage utilization     12. Control units         32. History data files*
3. SSI data menu*          13. I/O device menu*      33. Benchmark displays*
4. Priv. operations        14. Reserved              34. Correlation coeff.
5. System counters         15. Cache extend. func.*  35. System summary*
6. CP IUCV services        16. Reserved              36. Auxiliary storage
7. SPOOL file display*     17. DASD seek distance*   37. CP communications*
8. LPAR data menu*         18. I/O prior. queueing*  38. DASD load
```

```
9. Shared segments          19. I/O configuration      39. Minidisk cache*
A. Shared data spaces       1A. I/O config. changes    3A. Storage mgmt. data*
B. Virt. disks in stor.                                3B. Proc. load & config*
C. Transact. statistics     User Data                  3C. LPAR logs menu*
D. Monitor data             21. User resource usage*   3D. Response time (all)*
E. Monitor settings         22. User paging menu*      3E. RSK data menu*
F. System settings          23. User wait states*      3F. Scheduler queues
G. System configuration     24. User response time*    3G. Scheduler data
H. VM Resource Manager      25. Resources/transact.*   3H. SFS/BFS logs menu*
                            26. User communication*    3I. System log
I. Exceptions               27. Multitasking users*    3K. TCP/IP data menu*
                            28. User configuration*    3L. User communication
K. User defined data*       29. Linux systems*         3M. User wait states

            Pointers to related or more detailed performance data
            can be found on displays marked with an asterisk (*).
```

Drilling down into report panes

You can now use the active report panes. To drill down into these panes, move the cursor to any of the titles that are active. (Active titles display the number or letter in white; inactive titles are green.) The more useful report panes to drill down into include the following examples:

```
21. User resource usage
22. User paging load
23. User wait states
28. User configuration
29. Linux systems
33. Benchmark displays
```

6.3 Collecting and using raw CP monitor data

Although Performance Toolkit formats and displays current performance data, you also often must look at older data. Typically, you compare the current system performance to the past performance. Data must be available for troubleshooting or to generate reports.

6.3.1 Collecting CP monitor data

CP monitor records are collected by the **MONWRITE** utility and written to a disk or tape. The resulting file contains all of the original unprocessed data. This data can be used later to generate reports. Performance Toolkit can use this data in Monitor Data Scan Mode to look at historical data as though it were current.

Complete the following steps:

1. Log on to the **MONWRITE** virtual machine.
2. Edit the **PROFILE EXEC**:

```
LOGON MONWRITE
z/VM Version 6 Release 3.0, Service Level 0000 (64-bit),
built on IBM Virtualization Technology
There is no logmsg data
FILES:   NO RDR,   NO PRT,   NO PUN
LOGON AT 10:40:31 EDT FRIDAY 06/07/13
```

```
z/VM V6.3.0    2013-06-04 12:50

Ready; T=0.01/0.01 10:40:34
===> x profile exec a
input
/* ALL MONITOR COMMANDS ARE LOCATED IN AUTOLOG1'S PROFILE EXEC */
'MONWRITE MONDCSS *MONITOR DISK CLOSE 480'
===> file
```

3. Run the REXX exec that is named **profile**:

```
===> profile
HCPMOW6272I Now recording in file D060713 T110146 A1
HCPMOW6265A MONITOR WRITER CONNECTED TO *MONITOR
```

4. Disconnect from MONWRITE:

```
===> #cp disc
```

The **CLOSE 480** statement tells **MONWRITE** to close the output file every 8 hours (480 minutes), starting from midnight. Therefore, regardless of when it starts recording, it closes the file at 08:00, 16:00, and at 24:00. The file name clearly shows the date and time when the recording started.

To collect **MONWRITE** data automatically, start the **MONWRITE** virtual machine when you IPL z/VM. Add a line to the **PROFILE EXEC** of the AUTOLOG1 191 disk (or AUTOLOG2 191 if an external security manager, such as IBM RACF® is running):

```
===> x profile exec
...
"CP XAUTOLOG MONWRITE"              /* Start the MONWRITE VM       */
...
```

The **MONWRITE** A-disk is shipped as 300 cylinders, which is a small disk. Depending on the monitor interval activity of the system and the number of samples/events, it can fill quickly. When the disk is full, **MONWRITE** can no longer write.

It is important to monitor the space on **MONWRITE**'s A-disk. You can also use a utility that archives old files and cleans up the space automatically. MONCLEAN is an example of this type of a utility. You can download MONCLEAN from this web page.

Complete the following steps to install MONCLEAN:

1. Use FTP binary to transfer MONCLEAN VMARC to the MONWRITE 191 disk.
2. Run MONWRITE VMARC by using the **pipe** command:

```
===> pipe < monclean vmarc a | fblock 80 00 | > monclean vmarc A F 80
```

3. Unpack the MONCLEAN VMARC file by using the **VMARC** command:

```
===> vmarc unpk monclean vmarc a
MONCLEAN EXEC     A1. Bytes in=      4080, bytes out=      7678 (   188%).
MONCLEAN README   A1. Bytes in=      1040, bytes out=      2240 (   215%).
```

4. Check the documentation in MONCLEAN README.
5. Modify the PROFILE EXEC:

```
===> x profile exec
/* ALL MONITOR COMMANDS ARE LOCATED IN AUTOLOG1'S PROFILE EXEC */
'MONWRITE MONDCSS *MONITOR DISK CLOSE 60 EXEC MONCLEAN'
```

6. Start recording:

```
===> profile
HCPMOW6272I Now recording in file D061213 T131724 A1
HCPMOW6265A MONITOR WRITER CONNECTED TO *MONITOR
```

7. MONWRITE closes the output file every hour and run MONCLEAN EXEC. If the MONCLEAN EXEC was not modified, it removes the oldest file when the disk reaches 80% full.

   Example 6-2 shows the MONWRITE 191 disk when **MONCLEAN** is running.

Example 6-2 MONWRITE 191 disk

```
MAINT    FILELIST A0  V 169  Trunc=169 Size=19 Line=1 Col=1 Alt=0
Cmd   Filename Filetype Fm Format Lrecl     Records     Blocks   Date     Time
      D061313  T100016  Z1 F       4096       49275      49275  6/13/13 10:29:16
      D061313  T090016  Z1 F       4096       99407      99407  6/13/13 10:00:15
      D061313  T080015  Z1 F       4096       99392      99392  6/13/13  9:00:15
      D061313  T070015  Z1 F       4096       99348      99348  6/13/13  8:00:15
      D061313  T060015  Z1 F       4096       99348      99348  6/13/13  7:00:15
      D061313  T050016  Z1 F       4096       99348      99348  6/13/13  6:00:15
      D061313  T040016  Z1 F       4096       99348      99348  6/13/13  5:00:15
      D061313  T030015  Z1 F       4096       99348      99348  6/13/13  4:00:15
      D061313  T020016  Z1 F       4096       99348      99348  6/13/13  3:00:15
      D061313  T010015  Z1 F       4096       99348      99348  6/13/13  2:00:15
      D061313  T000015  Z1 F       4096       99348      99348  6/13/13  1:00:15
      D061213  T230015  Z1 F       4096       99348      99348  6/13/13  0:00:15
      D061213  T220015  Z1 F       4096       99356      99356  6/12/13 23:00:15
      D061213  T210015  Z1 F       4096       99357      99357  6/12/13 22:00:15
      D061213  T200015  Z1 F       4096       99348      99348  6/12/13 21:00:15
      PROFILE  EXEC     Z1 V         65           2          1  6/12/13 11:35:49
      MONCLEAN EXEC     Z1 V         75         194          2  6/12/13 11:32:13
      MONCLEAN README   Z1 F         80          28          1  6/12/13 11:32:13
      MONCLEAN VMARC    Z1 F         80          64          2  6/12/13 11:32:13
```

6.3.2 Using CP monitor data

You can use the Performance Toolkit subcommand **MONSCAN** to select a CP monitor file on disk or tape (that was created by the standard **MONWRITE** utility) as input for performance data analysis. When the specified file is found, a performance data scan mode is entered that looks almost identical to the normal real-time monitoring mode. You can use this mode to browse through the accumulated monitor data.

Because PERFSVM virtual machine is used to show the current performance data, it is better to use a different virtual machine to perform **MONSCAN**. The following example uses the MAINT user ID:

1. Link and access the PERFSVM 201 minidisk:

```
===> vmlink perfsvm 201
DMSVML2060I PERFSVM 201 linked as 0120 file mode Z
```

2. Link and access the MONWRITE 191 minidisk:

```
===> vmlink monwrite 191
DMSVML2060I MONWRITE 191 linked as 0121 file mode X
```

3. Check the files that are available from MONWRITE:

```
===> filel * * x
```

```
MAINT    FILELIST A0  V 169  Trunc=169 Size=4 Line=1 Col=1 Alt=0
Cmd   Filename Filetype Fm Format Lrecl    Records     Blocks   Date     Time
      DO61013  TO84824  X1 F       4096      53930      53930  6/10/13  9:20:43
      PROFILE  EXEC     X1 V         65          3          1  6/10/13  8:48:21
```

4. Run the MONSCAN subcommand:

```
===> perfkit monscan DO61013 TO84824 X
```

The Regular Performance Screen Selection panel appears.

```
FCX124          Performance Screen Selection  (FL630        )    Monitor Scan

General System Data        I/O Data                  History Data (by Time)
1. CPU load and trans.     11. Channel load          31. Graphics selection
2. Storage utilization     12. Control units         32. History data files*
3. SSI data menu*          13. I/O device menu*      33. Benchmark displays*
4. Priv. operations        14. Reserved              34. Correlation coeff.
5. System counters         15. Cache extend. func.*  35. System summary*
6. CP IUCV services        16. Reserved              36. Auxiliary storage
7. SPOOL file display*     17. DASD seek distance*   37. CP communications*
8. LPAR data               18. I/O prior. queueing*  38. DASD load
9. Shared segments         19. I/O configuration     39. Minidisk cache*
A. Shared data spaces      1A. I/O config. changes   3A. Storage mgmt. data*
B. Virt. disks in stor.                              3B. Proc. load & config*
C. Transact. statistics    User Data                 3C. Logical part. load
D. Monitor data            21. User resource usage*  3D. Response time (all)*
E. Monitor settings        22. User paging load*     3E. RSK data menu*
F. System settings         23. User wait states*     3F. Scheduler queues
G. System configuration    24. User response time*   3G. Scheduler data
H. VM Resource Manager     25. Resources/transact.*  3H. SFS/BFS logs menu*
                           26. User communication*   3I. System log
I. Exceptions              27. Multitasking users*   3K. TCP/IP data menu*
                           28. User configuration*   3L. User communication
K. User defined data*      29. Linux systems*        3M. User wait states

            Pointers to related or more detailed performance data
            can be found on displays marked with an asterisk (*).
```

5. Select an option; for example, 1 - CPU Load. The first panel does not contain any data. Enter the **nexts** command (next sample) and a panel with actual numbers appears. You can see the interval on the top of the panel:

```
FCX100      Data for 2013/06/10  Interval 08:48:40 - 08:49:40    Monitor Scan

CPU Load                                                  Status or
PROC TYPE %CPU  %CP %EMU %WT %SYS %SP %SIC %LOGLD  ded. User
P00  CP      0    0    0 100    0   0   99      0  Master
P01  CP      0    0    0 100    0   0   99      0  Alternate
P02  IFL     0    0    0 100    0   0  ...      0  Alternate
P03  IFL     0    0    0 100    0   0  ...      0  Alternate

Total SSCH/RSCH    254/s     Page rate        .0/s     Priv. instruct.    28/s
Virtual I/O rate    10/s     XSTORE paging    .0/s     Diagnose instr.    16/s
Total rel. SHARE   3050     Tot. abs SHARE     0%

Queue Statistics:     Q0     Q1     Q2     Q3     User Status:
VMDBKs in queue        1      0      1      0     # of logged on users      14
VMDBKs loading         0      0      0      0     # of dialed users          0
```

```
Eligible VMDBKs               0      0      0     # of active users          7
El. VMDBKs loading            0      0      0     # of in-queue users        2
Tot. WS (pages)     2911      0  41870      0     % in-Q users in PGWAIT     0
Reserved                                           % in-Q users in IOWAIT     0
85% elapsed time   96.00  16.00  128.0  768.0     % elig. (resource wait)    0

Transactions      Q-Disp    trivial   non-trv     User Extremes:
Average users       2.7         .8        .2     Max. CPU %   LNXADMIN     .1
Trans. per sec.      .2         .1        .0     Reserved
Av. time (sec)    18.40      12.39     16.39     Max. IO/sec  MONWRITE    9.4
UP trans. time                .000      .000     Max. PGS/s   ........  .....
MP trans. time               12.39     16.39     Max. RESPG   LNXADMIN  41923
System ITR (trans. per sec. tot. CPU)    31.3     Max. MDCIO   MONWRITE     .1
Emul. ITR (trans. per sec. emul. CPU)   269.2     Max. XSTORE  ........  .....
```

6.4 Monitoring Linux performance and troubleshooting

Previous sections described how the Performance Toolkit can show resource consumption of the Linux guest as measured and dispatched by the z/VM hypervisor. z/VM is not aware of the nature of the guest and it cannot understand what is happening inside the guest. For that reason, it is important to be able to measure performance data from within the Linux guest itself. To monitor Linux performance data at this level, a data gatherer process must be running within each Linux guest that you want to monitor.

Different methods exist of gathering this data. Many commercial and non-commercial solutions are available for long-term monitoring, also. This book cannot cover all of the requirements for long-term monitoring (low CPU consumption, data storage, and similar). This chapter shows how to monitor Linux performance in short periods, especially when you are troubleshooting performance problems.

6.4.1 Monitoring Linux performance from z/VM

This section describes how to gather Linux performance data and provide this data to z/VM for a consolidated overview.

To monitor Linux performance data directly from the kernel, the following prerequisites must be met:

- The `APPLMON` option is set in the user directory.
- `APPLMON` data monitoring is built into the kernel. This feature is built into Red Hat Enterprise Linux.

Complete the following steps to use this built-in monitoring function:

1. Start a Secure Shell (SSH) session to a Linux system. In our example, `LINUX3` is used.
2. Three modules are built into the kernel, but they are not loaded by default. They are named `appldata_mem`, `appldata_os`, and `appldata_net_sum`. You can verify that they are not loaded with the **`lsmod`** and **`grep`** commands:

   ```
   # lsmod | grep appldata
   ```

3. If no output results, no modules with the string `appldata` loaded are available. Load those modules by using the **`modprobe`** command and verify that they were loaded:

   ```
   # modprobe appldata_mem
   ```

```
# modprobe appldata_os
# modprobe appldata_net_sum
```

4. If you repeat the **lsmod** command, you see the following output:

```
# lsmod | grep appldata
appldata_net_sum       1966  0
appldata_os            2989  0
appldata_mem           2008  0
```

The monitoring variables are in the /proc/sys/appldata/directory. The following five files are in this directory:

timer	Controls whether any data gathering is in effect
interval	Sets the interval, in milliseconds, in which samples will be taken
mem	Controls the memory data gathering module
os	Controls the CPU data gathering module
net_sum	Controls the net data gathering module

5. To turn on the built-in kernel monitoring, use the **echo** command to send a nonzero value into four of the five monitoring variables in the /proc/ virtual file system:

```
# echo 1 > /proc/sys/appldata/timer
# echo 1 > /proc/sys/appldata/mem
# echo 1 > /proc/sys/appldata/os
# echo 1 > /proc/sys/appldata/net_sum
```

Built-in kernel monitoring is now turned on. Run the monitoring only for specific periods. As Linux monitoring data is captured, the Performance Toolkit's minidisk space fills quickly.

Viewing performance data from the Linux kernel in Performance Toolkit

After the system collects data for a period, use the Performance Toolkit to view Linux performance data. To view that data, drill down into menu 29, Linux systems, either from the browser interface or the 3270 interface, as shown in Example 6-3.

Example 6-3 Linux display

```
FCX242      CPU 2817  SER 23BD5          Linux Displays         Perf. Monitor


   Linux screens selection
 S Display     Description
 . LINUX       RMF PM system selection menu
 . LXCPU       Summary CPU activity display
 S LXMEM       Summary memory util. & activity display
 . LXNETWRK    Summary network activity display
```

Then, enter S over the period on the left side of the submenu panel in the row that corresponds to the report that you want to see. You will see a new report panel with the Linux guest systems memory overview, as shown in Example 6-4.

Example 6-4 Linux guests memory overview

```
FCX244      CPU 2817  SER 23BD5        Initial  14:22:57         Perf. Monitor
______           .      .       .      .      .       .      .       .      .
          <------------ Memory Allocation (MB) -------------> <------- Swapping
Linux     <--- Main ---> <--- High --->        Buffers  Cache <-Space (MB)-> <-
Userid    M_Total %MUsed H_Total %HUsed Shared /CaFree   Used S_Total %SUsed
>System<    491.6   25.8      .0     .0     .0     8.6   46.3   761.6     .0
```

```
LINUX3      491.6   25.8      .0      .0      .0      8.6    46.3   761.6      .0
```

You can also use a web interface to view the same data.

6.4.2 Monitoring Linux performance from inside Linux

Many tools are available for Linux performance monitoring. This section describes several commonly used tools. They are all platform independent and they work on Linux in general.

top command

When you run the **top** command without any parameters, it shows a system overview and the tasks that are running are similar to Example 6-5. The output is refreshed every 3 seconds automatically. To leave the **top** command, press **q**.

Example 6-5 top command

```
top - 06:48:26 up 8 days, 15:00,  1 user,  load average: 0.00, 0.00, 0.00
Tasks: 126 total,   1 running, 125 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.1 us,  0.2 sy,  0.0 ni, 99.8 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  15945.4 total,  14662.3 free,    537.5 used,    745.7 buff/cache
MiB Swap:    761.6 total,    761.6 free,      0.0 used.  15207.7 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   5526 root      20   0   35696   8544   6808 S   0.3   0.1   4:13.30 pmdaproc
      1 root      20   0  104728  14136   8860 S   0.0   0.1   0:13.32 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.11 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
     10 root      20   0       0      0      0 S   0.0   0.0   0:00.44 ksoftirqd/0
...
```

vmstat command

Another useful command is the **vmstat** command, which reports information about processes, memory, paging, block IO, traps, and CPU activity. When you run **vmstat** without any parameters, it shows one line that summarizes averages since the last IPL, which is not useful.

Example 6-6 shows the **vmstat 5** output. This version of the command shows the first line with averages since the last IPL and then writes a new line every 5 seconds with the current data, which is more useful.

Example 6-6 vmstat 5 command output

```
# vmstat 5
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 666832   4596 239924    0    0     4     1   79  126  0  0 100  0  0
 0  0      0 666768   4596 239956    0    0     0     0  149  248  0  0 100  0  0
 0  0      0 666752   4596 239956    0    0     0     0  149  246  0  0 100  0  0
 0  0      0 666720   4596 239956    0    0     0     0  171  266  0  0 100  0  0
 0  0      0 666752   4596 239956    0    0     0     0  169  258  0  0 100  0  0
 0  0      0 666752   4596 239956    0    0     0     0  166  254  0  0 100  0  0
 0  0      0 666752   4596 239956    0    0     0     0  160  247  0  0 100  0  0
 0  0      0 666720   4596 239956    0    0     0     0  152  250  0  0 100  0  0
```

The `wa` column shows a wait time and represents a percentage of time while the system waited for I/O. The higher the percentage, the more time that tasks waste nonproductively.

The `st` column shows *stolen time*. It represents the time that the CPU was stolen from a guest by the hypervisor. This stolen time can mean several things: CPU contention at the z/VM level, heavy z/VM paging, heavy virtual switch usage, and so on. The higher the number, the more time that a guest will spend nonproductively.

The sysstat suite of resource monitoring tools

In this section, we describe tools that are part of the `sysstat` package, which might not be installed automatically. Check whether the package is installed when you use Red Hat Enterprise Linux 7.1, and install it if it is not installed.

Complete the following steps:

1. Check whether the sysstat package is installed:

```
# rpm -qa | grep sysstat
sysstat-11.7.3-2.el8.s390x
```

2. If it is not installed, install it by using the command:

```
# yum install sysstat
...
Installed:
  sysstat.s390x 0:11.7.3-2.el8

Dependency Installed:
  lm_sensors-libs.s390x 0:3.4.0-21.el8
```

iostat

In addition to reporting overall system performance, the **iostat** command provides information about input/output device loading. The **iostat** command reports the time that the devices are active in relation to their average transfer rates. Example 6-7 shows a sample output of the **iostat -x 5 2** command. The **-x** is the command to display extended statistics, **5** is the interval in seconds, and **2** is the number of reports. (Display extended statistics for 2 reports at 5 second intervals.)

Example 6-7 iostat command output

```
Linux 4.18.0-193.el8.s390x (rdbkrh10.pbm.ihost.com)     09/24/2020      _s390x_ (4 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.04    0.01    0.03    0.00    0.01   99.92

Device            r/s     w/s     rkB/s     wkB/s   rrqm/s   wrqm/s  %rrqm  %wrqm r_await w_await aqu-sz
rareq-sz wareq-sz  svctm  %util
dasda            0.02    0.27      0.77      8.98     0.00     0.03   1.01   9.31    0.46    0.88   0.00
42.79    33.62   3.64   0.10
dm-0             0.02    0.29      0.69      8.96     0.00     0.00   0.00   0.00    0.42    0.23   0.00
39.92    30.40   3.32   0.10
dasdb            0.00    0.00      0.01      0.00     0.00     0.00   0.00   0.00    0.53    0.00   0.00
24.85     0.00   0.64   0.00
dasdc            0.00    0.00      0.01      0.00     0.00     0.00   0.00   0.00    0.56    0.00   0.00
24.94     0.00   0.72   0.00
loop0            0.00    0.00      0.02      0.00     0.00     0.00   0.00   0.00    0.91    0.00   0.00
83.31     0.00   1.61   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
```

```
           0.00    0.00    0.00    0.00    0.00  100.00

Device            r/s     w/s     rkB/s     wkB/s   rrqm/s   wrqm/s  %rrqm  %wrqm r_await w_await aqu-sz
rareq-sz wareq-sz  svctm  %util
dasda            0.20    0.00      1.60      0.00     0.00     0.00   0.00   0.00    0.00    0.00   0.00
8.00     0.00    0.00   0.00
dm-0             0.20    0.00      1.60      0.00     0.00     0.00   0.00   0.00    0.00    0.00   0.00
8.00     0.00    0.00   0.00
dasdb            0.00    0.00      0.00      0.00     0.00     0.00   0.00   0.00    0.00    0.00   0.00
0.00     0.00    0.00   0.00
dasdc            0.00    0.00      0.00      0.00     0.00     0.00   0.00   0.00    0.00    0.00   0.00
0.00     0.00    0.00   0.00
loop0            0.00    0.00      0.00      0.00     0.00     0.00   0.00   0.00    0.00    0.00   0.00
0.00     0.00    0.00   0.00
```

The `aqu-sz` column shows the average queue size for a specific device. The larger the number, the more contention exists for that device. This field was known as `avgqu-sz` in previous versions.

The *await* column displays average wait time for a device. It includes the time that requests spent in the queue and the time that was spent servicing requests in a storage device. The higher the number, the more time is wasted by a program that waits for I/O.

sar and sadc commands

The **sadc** command stores data in a binary file. The command is used to provide data to the **sar** command. Data is gathered by calling the data collector by using the **sadc** command. In this example, all data for all activities will be collected 60 times in 5-second intervals (5 x 60 = 300 s = 5 minutes):

```
# /usr/lib64/sa/sadc -S ALL -F 5 60 /tmp/sadc.out
```

The output that is produced by the **sadc** command is a binary file. The **sar** command is used to process the binary file and generate text output. The following example displays all of the data that is collected from the file `/tmp/sadc.output` and writes the results to a file called `output.txt`:

```
# sar -A -f /tmp/sadc.output > outfile.txt
```

The **sar** command generates detailed performance information.

If **sar** is configured as a service and gathers data automatically, its data is stored in the `/var/log/sa` directory. Data files are of the form `sa<dd>` where `dd` is the day of the month. Text files are of the form `sar<dd>`, for example:

```
# cd /var/log/sa
# file *
sa09:  data
sa10:  data
sa11:  data
sa12:  data
sa13:  data
sar09: ASCII text
sar10: ASCII text
sar11: ASCII text
sar12: ASCII text
```

6.5 Red Hat Insights

Red Hat Insights is a cloud service that helps proactive monitoring and analysis of Red Hat Enterprise Linux systems. It is offered for free as part of Red Hat Enterprise Linux subscriptions.

Insights gathers information from each client and reports that are problems related to configuration, performance, security, and compliance. It also provides recommendations to mitigate and resolve those problems.

Insights can be integrated with Ansible to generate and run playbooks to address the problems on a single system or group of systems together.

Custom policies can be defined in Insights to compare with the registered systems and notify in case of deviation from those policies.

6.5.1 Installing and registering the client

Each RHEL system must be subscribed to a valid Red Hat subscription to access the Red Hat Insights cloud service. For more information about subscribing your RHEL system with a valid entitlement, see 3.1, “Subscribing to a Red Hat Enterprise system by using subscription-manager” on page 46.

Complete the following steps to install Red Hat Insights on the system:

1. Use yum to install the Insights client package, as shown in Example 6-8.

Example 6-8 Installing Insights

```
===> yum install insights-client
Updating Subscription Management repositories.
Red Hat Enterprise Linux 8 for IBM z Systems - BaseOS (RPMs)              3.0
MB/s |  16 MB     00:05
Red Hat Enterprise Linux 8 for IBM z Systems - AppStream (RPMs)            48
MB/s |  15 MB     00:00
Package insights-client-3.0.13-1.el8.noarch is already installed.
Dependencies resolved.
================================================================================
Package                   Architecture     Version
Repository                                                                 Size
================================================================================
Upgrading:
 insights-client           noarch           3.0.13-1.el8_1
rhel-8-for-s390x-appstream-rpms   907 k

Transaction Summary
================================================================================
==================================
Upgrade  1 Package

Total download size: 907 k
Is this ok [y/N]: y
Downloading Packages:
insights-client-3.0.13-1.el8_1.noarch.rpm                787 kB/s | 907 kB
00:01
```

```
--------------------------------------------------------------------------------
----------------------------------Total                   787 kB/s | 907 kB
00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                1/1
  Running scriptlet: insights-client-3.0.13-1.el8_1.noarch            1/1
  Upgrading        : insights-client-3.0.13-1.el8_1.noarch            1/2
  Running scriptlet: insights-client-3.0.13-1.el8_1.noarch            1/2
  Running scriptlet: insights-client-3.0.13-1.el8.noarch              2/2
  Cleanup          : insights-client-3.0.13-1.el8.noarch          2/2
  Running scriptlet: insights-client-3.0.13-1.el8.noarch              2/2
  Running scriptlet: insights-client-3.0.13-1.el8_1.noarch            2/2
  Running scriptlet: insights-client-3.0.13-1.el8.noarch              2/2
  Verifying        : insights-client-3.0.13-1.el8_1.noarch            1/2
  Verifying        : insights-client-3.0.13-1.el8.noarch              2/2
Installed products updated.

Upgraded:
  insights-client-3.0.13-1.el8_1.noarch

Complete!
```

2. Register the client to the Insights cloud service and start data gathering, as shown in Example 6-9.

Example 6-9 Registering the client

```
===> insights-client --register
You successfully registered 7a54af82-34ba-4aef-937c-fbc32718df5f to account
6256786.
Successfully registered host rdbkrh10.pbm.ihost.com
Automatic scheduling for Insights has been enabled.
Starting to collect Insights data for rdbkrh10.pbm.ihost.com
Uploading Insights data.
Successfully uploaded report from rdbkrh10.pbm.ihost.com to account 6256786.
View the Red Hat Insights console at https://cloud.redhat.com/insights/
```

3. Log on to `https://cloud.redhat.com/insights/` by using your Red Hat account to view the Red Hat Insights dashboard.

6.5.2 Insights dashboard

After you register your client, you can log on to the Red Hat Insights dashboard at `https://cloud.redhat.com/insights/` by using your Red Hat account. You are presented with the window that is shown in Figure 6-3.

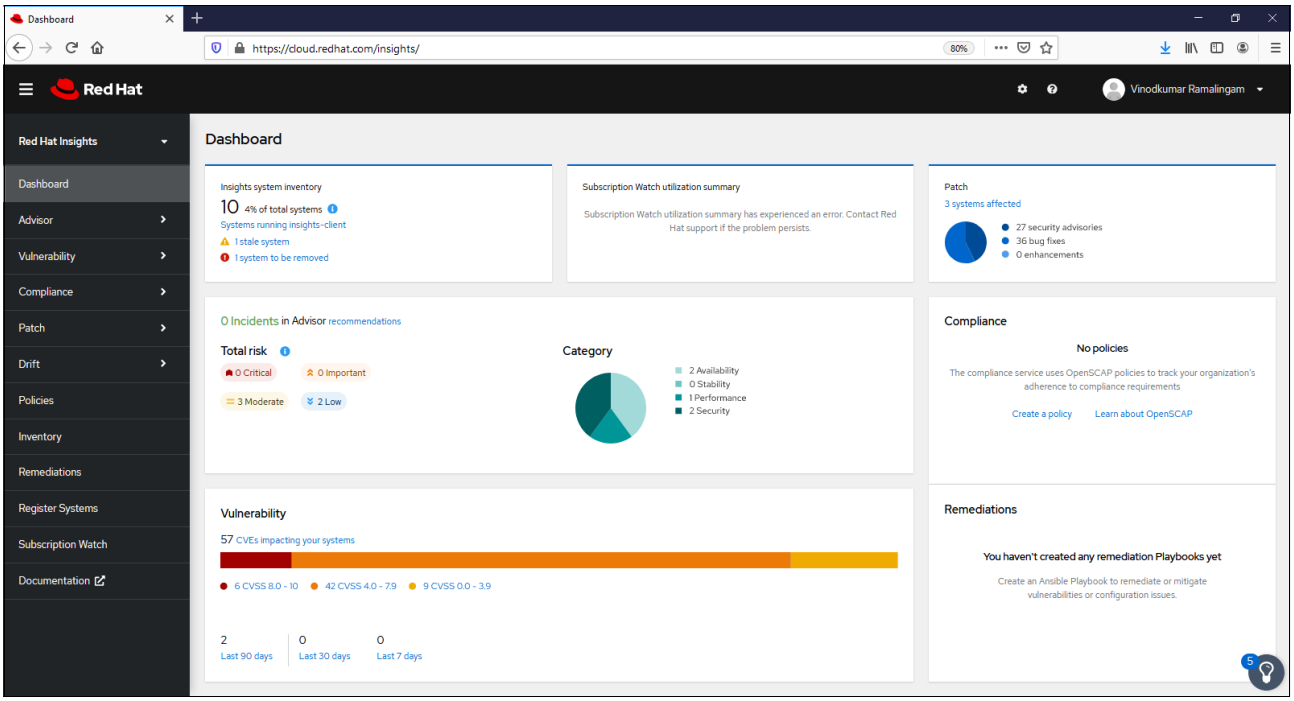

Figure 6-3 Red Hat Insights Dashboard

The dashboard provides a summary of security and compliance status of all your registered systems. This group of registered systems in Insights is called as the *Insights System Inventory*. The Dashboard consists of tiles to represent statuses of the systems in inventory such as risks, security advisories, vulnerabilities, compliance, and remediation.

The left panel of the Insights dashboard helps navigate to various sections, such as Advisories, Vulnerability, and Compliance. It also lets you define a baseline against which the systems can be compared between each other and how they drift from the baselines.

You can register new systems and access documentation. Subscription Watch lets you track your Red Hat Subscriptions from the dashboard.

Go to Inventory and select the system that you registered to view the details, as in Figure 6-4.

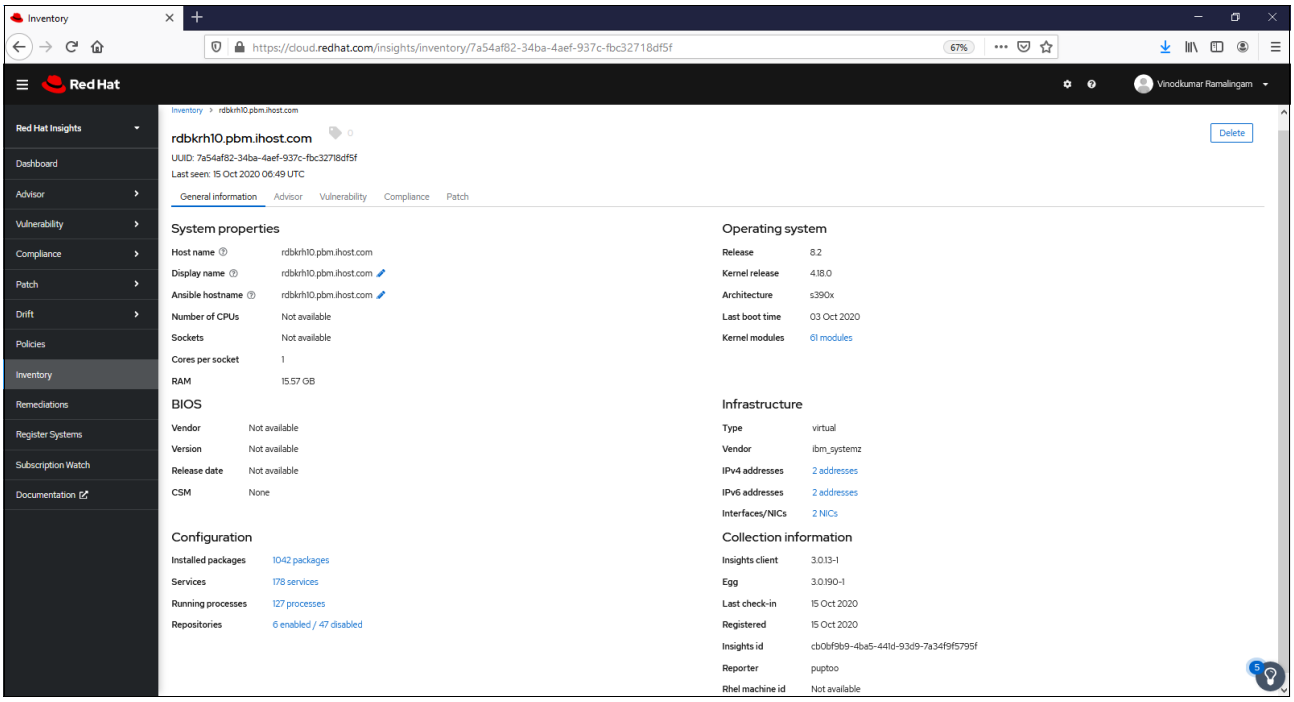

Figure 6-4 Red Hat Insights Inventory

Red Hat Insights now offers the following services:

- Advisor: Identifies known configuration risks in the operating system, underlying infrastructure, or workloads that affect performance, stability, availability, or security best practices
- Vulnerability: Assesses, remediates, and reports on CVEs that affect Red Hat Enterprise Linux environments in the cloud or on-premises.
- Compliance: Analyzes the level of compliance of a Red Hat Enterprise Linux environment to an OpenSCAP policy, based on the Red Hat Security guide.
- Patch: Determines which Red Hat product advisories apply to an organization's specific Red Hat Enterprise Linux instances. It provides guidance for manual remediation (manually or by using Ansible Playbooks for patching).
- Drift compares systems to baselines, system histories, and to each other to troubleshoot or identify differences.
- Policies enable organizations to define and monitor for policies that are important internally with alerts for environments that are not aligned to a policy.

7

Configuring Linux for cloning

Cloning is a process where new systems are created from a master system, which is also called the *golden image*. This chapter simplifies the process of cloning when it is compared to the previous releases of this book. The cloning procedure benefits from several improvements to the z/VM environment, which simplifies the cloning process.

Linux operating systems now have more unique identifiers. For example, with the introduction of systemd, a new machine ID is added. All of these identifiers must be re-created on the cloned system. However, the process to know all of these identifiers and the process to re-create them require in-depth knowledge of the golden image. Failure to update all of these identifiers can cause unexpected problems later, including data corruption or security issues.

If you are unsure of all of the unique identifiers for your golden image, and you prefer not to follow the cloning process, see the automated installation procedures in Chapter 2, “Automated Red Hat Enterprise Linux installations by using Kickstart” on page 35.

This chapter describes an example of the cloning process. A basic cleanup procedure is supplied on a best-effort basis. You must review the cleanup script. The cleanup script will require updates whenever a more specialized golden image needs to be prepared.

After the cleanup, the configuration update of the target system was prepared as an example. Depending on the services that are used, the configuration update of the target system also will need updates from the administrator. You must review the configuration update of the target system.

This chapter includes the follow topics:

7.1 Creating a golden image for cloning

A golden image is a special type of Linux installation that is used as a baseline to generate new Linux systems from it.

Important: Be careful when you prepare the golden image. All unique identifiers must be updated for the resulting new Linux system, which is known as the *cloned system*.

Before you start this configuration, ensure that all of the files that are associated with this book are available. These additional materials are in Appendix B, “Additional material” on page 241.

In this book, we assume that the user name of the z/VM guest is `GOLD`. Ensure that you make all necessary changes that you want to for your template system before you proceed.

To prepare the Linux system as the golden image, complete the following steps:

1. Copy the prepare script to the system. It removes files that are specific to one installation:

   ```
   # scp prepare_system.sh golden.itso.ibm.com:/usr/local/bin
   ```

   Note: When you copy the script files to your Linux system, make sure they do not contain any bad characters if copied or transferred from other platforms or over FTP. To avoid such problems use the following command to convert the file to the proper character-set:

   ```
   # dos2unix prepare_system.sh
   dos2unix: converting file prepare_system.sh to Unix format...
   ```

2. Log on to the golden image and ensure that the scripts under `/usr/local/bin` are executable:

   ```
   # ssh golden.itso.ibm.com
   # chmod 755 /usr/local/bin/*
   ```

3. Shut down the image by calling the **`prepare_system.sh`** script:

   ```
   # /usr/local/bin/prepare_system.sh

   Warning: This utility will prepare the root disk for cloning by removing some
            configuration files. After it has finished it will halt this systemd.

   press Ctrl+C within the next 15 seconds to abort

   ln -s '/etc/systemd/system/clone.service'
   '/etc/systemd/system/network.target.wants/clone.service'
   Removing SSH keys
   Removing the network configuration
   Removing this script (/usr/local/bin/prepare_system.sh)
   halt the system
   ```

4. Make a backup of the image.

7.2 Cloning the golden image by using DirMaint

Use IBM Directory Maintenance (DirMaint) for z/VM to perform the copy of the golden image to a new user. To clone the golden image to a user with user name LINUX6, complete the following steps:

1. Ensure that the necessary resources, such as disks, are prepared. The necessary tasks are described in Chapter 6, “Planning and preparation for Linux workloads”, of *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, hSG24-8147.
2. Log off the source system (golden).
3. Log on as MAINT.
4. If the z/VM guest is not created, use the **dirmaint** command to create it:

   ```
   ==> dirmaint add LINUX6 like LNXPROTO pw <NewPassword>
   ```

5. Ensure that the source and target systems are logged off:

   ```
   ==> query linux5 #query linux6
   query linux5
   query linux6
   HCPCQU045E LINUX5 not logged on
   Ready(00045); T=0.01/0.01 10:11:14
   HCPCQU045E LINUX6 not logged on
   Ready(00045); T=0.01/0.01 10:11:14
   ```

6. Copy the disk 0100 from the source to the target system:

   ```
   ===> dirmaint for linux6 clonedisk 0100 linux5 0100
   dirmaint for linux6 clonedisk 0100 linux5 0100
   DVHXMT1191I Your CLONEDISK request has been sent for processing to
   DVHXMT1191I DIRMAINT at ITSOZVM1 via DIRMSAT2.
   Ready; T=0.01/0.01 08:41:24
    DVHREQ2288I Your CLONEDISK request for LINUX6 at
    DVHREQ2288I * has been accepted.
   ...
    DVHBIU3428I Changes made to directory entry
    DVHBIU3428I DATAMOV2 have been placed online.
    DVHSHN3430I CLONEDISK operation for LINUX6
    DVHSHN3430I address 0100 has finished (WUCF
    DVHSHN3430I 01085901).
   ```

7. Log off MAINT.

7.3 Sending the configuration update to the cloned system

Before the cloned system can be operational, the cloned system needs information about its configuration. Send a configuration script to the READER of the target system, LINUX6.

For Red Hat Enterprise Linux, this configuration script is called **send_config_rhel.sh**. To simplify the configuration, it is sufficient to create a small configuration file, which looks similar to Example 7-1.

Example 7-1 Sample configuration file to update network configuration of LINUX6

```
GENERATE_MACHINEID="1"
Hostname="LINUX6"
HostIP="9.12.7.103/24"
Gateway="9.12.4.1"
ReadChannel=0.0.0600
WriteChannel=0.0.0601
DataChannel=0.0.0602
ReadChannelSfx=600
Domain=itso.ibm.com
Nameserver=9.12.6.6
Layer2=1
```

Complete the following steps:

1. Log in as root on LNXADMIN.
2. Copy the **send_config_rhel.sh** to /usr/local/bin:

   # cp send_config_rhel.sh /usr/local/bin
3. Ensure that the scripts are executable:

   # chmod +x /usr/local/bin/send_config*
4. Create a configuration subdirectory to /root:

   # cd /root
   # mkdir /root/cnf
5. Create and adapt the additional configuration script as shown in Example 7-1 on page 144. The file name must match the USERID of the cloned system.

   # vi /root/cnf/LINUX6
6. Run the **send_config** script with the target USERID as the argument:

   RHEL: # send_config_rhel.sh LINUX6

Note: Make sure to use the dos2unix command for **send_config_rhel.sh** and **LINUX6** if they were transferred from different systems:

```
# dos2unix send_config_rhel.sh
dos2unix: converting file send_config_rhel.sh to Unix format...
# dos2unix LINUX6
dos2unix: converting file LINUX6 to Unix format...
```

The configuration update script is sent to the reader of LINUX6. Consider the following characteristics and requirements:

- File name that is sent to the reader is CLONE.
- File mode that is sent to the reader is SYSCTL.
- File must be sent by LNXADMIN.
- Lines of the script must not exceed 80 characters.

The script that is sent to LINUX6 is run during the first boot.

7.4 IPLing the cloned system

IPL the cloned system to activate it. You can be logged on, or if the system was set up with a common PROFILE EXEC that IPLs the system disk, a remote user with XAUTOLOG:

```
===> ipl 100
```

During the IPL, the systemd **clone.service** reads and runs the **clone.sysctl** script from the reader. The **clone.sysctl** script contains the information that is needed to finalize the target system.

Finally, restart the system (after the IPL) to ensure that the new machine-ID is used consistently with the cloned system.

8

Working with systemd

In modern Linux installations, systemd plays a major role. Understanding the concepts and knowing the utilities that are provided by systemd are key to any Linux administrator.

This chapter describes how to work with Red Hat Enterprise Linux 8.2 and systemd. It includes the following topics:

8.1 Getting started with systemd

Systemd is a system and services manager that replaced **upstart** in Red Hat Enterprise Linux. Systemd is the first user space process that the kernel starts when booting. This process is responsible for starting all of the services and their dependencies that allow the system to act as a server. Systemd uses units that can depend on other units.

The following types of units are available:

- Service: Used to start services
- Socket: Allows socket-based activations
- Device: Triggers reactions for devices as they appear or disappear
- Mount point: Controls mount points
- Target: Allows the grouping of units to act as synchronization points

For example, before the **local-fs.target** is reached, the local mount units, which depend on the device units, must be activated. By default, the system boots into the default target that pulls in all of the dependencies that are necessary to start the system.

For more information about the boot process, see 8.4, “System boot process” on page 156.

The units are described in unit files in `/etc/systemd/system` and `/usr/lib/systemd/system` where systemd searches for and loads them. Specific unit files are generated dynamically by systemd unit generators. For example, the **systemd-fstab-generator** parses the `/etc/fstab` and creates mount units for the entries, if necessary. RPM packages that provide services usually install their service unit files at `/usr/lib/systemd/system`.

Example 8-1 shows the `/usr/lib/systemd/system/sshd.service` unit file.

Example 8-1 Service unit file

```
[Unit]
Description=OpenSSH server daemon
Documentation=man:sshd(8) man:sshd_config(5)
After=network.target sshd-keygen.target
Wants=sshd-keygen.target

[Service]
Type=notify
EnvironmentFile=-/etc/crypto-policies/back-ends/opensshserver.config
EnvironmentFile=-/etc/sysconfig/sshd
ExecStart=/usr/sbin/sshd -D $OPTIONS $CRYPTO_POLICY
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartSec=42s

[Install]
WantedBy=multi-user.target
```

8.2 Using systemd units

This section describes how to manage services and isolate systemd targets.

8.2.1 Managing services

In this section, we describe the tasks that are used to manage the services with systemd. We use the systemctl tool to examine and control the systemd system and service manager.

Note: The systemctl utility does not communicate with services that were not started by systemd.

Listing active service units (running services)

Because systemd starts all of the services, you must know how to get a list of the available services. Use the **systemctl** command to control services and inspect their state:

```
# systemctl -t service
```

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
atd.service	loaded	active	running	Job spooling tools
auditd.service	loaded	active	running	Security Auditing Service
chronyd.service	loaded	active	running	NTP client/server
crond.service	loaded	active	running	Command Scheduler
...				

Note: The **systemctl** command attempts to fit the contents of its output into your terminal window. If the output has more lines than your terminal window can display, the **systemctl** command pipes the output into `less`, which is the default pager. You can change that behavior by adding the **--no-pager** option. If the width of your terminal window is too narrow to display the full contents, you can add the **--full** or **-l** option to show the full contents. The **--no-pager** and the **--full** options can be combined, if you want.

Listing failed service units

Use the **systemctl** command to list the failed service units:

```
# systemctl -t service --state=failed
```

Querying the status of a service unit

Use the **systemctl** command to query the state of a service unit:

```
# systemctl status sshd.service
sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2020-09-16 13:37:22 EDT; 2 days ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 1056 (sshd)
    Tasks: 1 (limit: 101851)
   Memory: 7.4M
   CGroup: /system.slice/sshd.service
           ‥1056 /usr/sbin/sshd -D
-oCiphers=aes256-gcm@openssh.com,chacha20-poly1305@openssh.com,aes256-ctr,aes256-cb>

Sep 16 13:37:22 rdbkrh04.pbm.ihost.com systemd[1]: Starting OpenSSH server daemon...
Sep 16 13:37:22 rdbkrh04.pbm.ihost.com sshd[1056]: Server listening on 0.0.0.0 port 22.
```

```
Sep 16 13:37:22 rdbkrh04.pbm.ihost.com sshd[1056]: Server listening on :: port 22.
Sep 16 13:37:22 rdbkrh04.pbm.ihost.com systemd[1]: Started OpenSSH server daemon.
Sep 16 13:38:07 rdbkrh04.pbm.ihost.com sshd[9087]: Accepted password for root from
9.160.39.131 port 53529 ssh2
Sep 16 13:38:07 rdbkrh04.pbm.ihost.com sshd[9087]: pam_unix(sshd:session): session opened
for user root by (uid=0)
Sep 19 04:50:14 rdbkrh04.pbm.ihost.com sshd[164505]: Accepted password for root from
9.163.26.123 port 50483 ssh2
Sep 19 04:50:14 rdbkrh04.pbm.ihost.com sshd[164505]: pam_unix(sshd:session): session opened
for user root by (uid=0)
...
```

In addition to the information that you expect, the output shows the last 10 log messages from the journal. If you only want to know whether a service unit is active, run the following command:

```
# systemctl is-active sshd.service
active
```

Stopping, starting, and restarting a service unit

Use the **systemctl** command to stop, start, and restart a service unit:

```
# systemctl stop vsftpd.service
# systemctl start vsftpd.service
# systemctl restart vsftpd.service
```

You can also omit the suffix and specify multiple units:

```
# systemctl restart vsftpd sshd
```

As a Linux administrator, you might change the configuration of a running service and want to reload its configuration without restarting the service. Service unit files can use the `ExecReload=` option to specify a command line (similar to sending a signal) that triggers the reload of the configuration. If your service supports it, you can trigger reload by using the following command:

```
# systemctl reload sshd
```

Not all services support that command:

```
# systemctl reload vsftpd
Failed to reload vsftpd.service: Job type reload is not applicable for unit
vsftpd.service.
```

To reload the configuration of services that support it and restart services that do not, run the following command:

```
# systemctl reload-or-restart sshd vsftpd
```

Listing the installed service units

Use the **systemctl** command to see a list of installed service units:

```
# systemctl list-unit-files -t service
UNIT FILE                                  STATE
arp-ethers.service                         disabled
atd.service                                enabled
auditd.service                             enabled
autovt@.service                            enabled
```

Listing the enabled service units

Use the **systemctl** command to list the service units that are wanted or required by another unit, such as the multi-user.target. Run the following command:

```
# systemctl list-unit-files -t service --state=enabled
UNIT FILE                                   STATE
atd.service                                 enabled
auditd.service                              enabled
autovt@.service                             enabled
chronyd.service                             enabled
cpi.service                                 enabled
```

Disabling a service

To disable starting a service at start time, use the **systemctl** command in the following form:

```
# systemctl disable vsftpd
Removed /etc/systemd/system/multi-user.target.wants/vsftpd.service.
```

Enabling a service

To configure a service to start automatically at boot time, use the following **systemctl** command:

```
# systemctl enable vsftpd
Created symlink /etc/systemd/system/multi-user.target.wants/vsftpd.service ·
/usr/lib/systemd/system/vsftpd.service.
```

Note: Issuing the **enable** command for a service unit does not trigger the activation of the service unit. It also does not start the service. Instead, it creates a dependency on another unit; for example, the `multi-user.target` unit. This dependency causes the start of the service whenever the unit that the service depends on is activated; for example, the multi-user.target. Disabling the service unit removes the dependency.

If you want to start and enable in one command, use the following command:

```
# systemctl enable --now vsftpd
Created symlink /etc/systemd/system/multi-user.target.wants/vsftpd.service ·
/usr/lib/systemd/system/vsftpd.service.
```

Checking the socket units that activate specific services

List the socket units in order of the listening address:

```
# systemctl list-sockets
LISTEN                              UNIT                            ACTIVATES
/run/dbus/system_bus_socket         dbus.socket                     dbus.service
/run/dmeventd-client                dm-event.socket
dm-event.service
/run/dmeventd-server                dm-event.socket
dm-event.service
...
```

8.2.2 Managing systemd target units

Target units group units and act as synchronization points. The concept of systemd targets is similar to the concept of the runlevels of the SysV-style init system, but a few differences exist. Instead of the use of runlevel numbers, the systemd targets use names because they are units. With systemd, multiple targets often are active because a target unit might depend on another target unit.

Listing active targets

Use the **`systemctl`** command to show a list of active targets:

```
# systemctl --type target
UNIT                   LOAD   ACTIVE SUB    DESCRIPTION
basic.target           loaded active active Basic System
cryptsetup.target      loaded active active Local Encrypted Volumes
getty.target           loaded active active Login Prompts
local-fs-pre.target    loaded active active Local File Systems (Pre)
local-fs.target        loaded active active Local File Systems
multi-user.target      loaded active active Multi-User System ...
```

Listing all targets

Use the **`systemctl`** command to display a list of all of the targets:

```
# systemctl --type target --all
UNIT                       LOAD      ACTIVE   SUB    DESCRIPTION
  basic.target               loaded    active   active Basic System
  cryptsetup.target          loaded    active   active Local Encrypted Volumes
  emergency.target           loaded    inactive dead   Emergency Mode
  getty-pre.target           loaded    inactive dead   Login Prompts (Pre)
  getty.target               loaded    active   active Login Prompts
  graphical.target           loaded    inactive dead   Graphical Interface
  initrd-fs.target           loaded    inactive dead   Initrd File Systems
  initrd-root-device.target loaded    inactive dead   Initrd Root Device
  initrd-root-fs.target      loaded    inactive dead   Initrd Root File System
...
```

Querying the default target that the system boots into

To determine the target unit that is used by default, run the following command:

```
# systemctl get-default
multi-user.target
```

Setting the default target into which the system boots

To configure the system to use a different default target unit, enter the following command at a shell prompt as `root`:

```
# systemctl set-default graphical.target
Created symlink /etc/systemd/system/default.target ·
/usr/lib/systemd/system/graphical.target.
```

Switching to a target

To turn off the GUI and change to the `multi-user.target` unit in the current session, run the following command as `root`:

```
# systemctl isolate graphical.target
```

The **isolate** command activates the specified target and all of its dependent units. All other units are stopped. By default, only a few targets can be isolated to prevent unusable system states. Table 8-1 maps important systemd targets to SysV runlevels.

Table 8-1 Mapping systemd targets to SysV runlevels

Systemd target	SysV runlevel	Notes
poweroff.target	0	Halts the system.
rescue.target	1, s, single	Single-user mode that provides a base system and a rescue shell.
multi-user.target	2, 3, 4	Multi-user, non-graphical but with network and services running.
graphical.target	5	Multi-user, graphical.
reboot.target	6	Reboot the system.
emergency.target	emergency	Emergency shell. This target is a special systemd target unit that can be specified as a kernel command-line argument: `systemd.unit=emergency.target`.

8.3 Working with the systemd journal

This section describes how to enable persistent journal data and how to view the journal.

8.3.1 Getting started with the journal

The journal is part of systemd and provides a modern logging mechanism. You can capture kernel log messages, regular syslog messages, the stdout/stderr that are written by services, and messages from the early boot stages. With the log message text, the journal stores metadata. The metadata consists of the process identifier (PID), user ID (UID), group ID (GID), executable, and so on. (For more information, enter the **man systemd.journal-fields** command on the Linux command line.) All of this information is indexed and can be queried by the administrator.

On Red Hat Enterprise Linux 8.2, the data of the journal is not stored permanently, by default. The systemd-journald is configured to store its data on a small in-memory disk (`/run/log/journal`, which is a tmpfs). Also, the systemd-journald is configured to forward the log messages to syslog, which means that they can be picked up by any traditional log daemon.

Both operating systems have the rsyslogd service that is enabled and that receives the messages from the journald and stores them persistently to `/var/log/messages`. Therefore, only the latest log data can be queried from the journal directly. To change the systemd-journald to store its data persistently on disk, create the `/var/log/journal` directory. When the journald is restarted, it detects the existence of this directory and starts to store the journal permanently on disk. Then, the traditional log daemon can be turned off.

Enabling persistent journal data at /var/log/journal

The systemd-journald is a system service that collects and stores logging data. It creates and maintains structured, indexed journals that are based on logging information that is received from various sources.

To enable it, run the following commands:

```
# mkdir /var/log/journal
# systemctl restart systemd-journald.service
```

Another way to enable persistent journal data is adding `Storage=persistent` into the `/etc/systemd/system.conf` file and recycling systemd by using the **`systemctl restart systemd-journald`** command. The use of this method ensures that systemd recreates the `/var/log/journal` directory, even if it is deleted by someone.

Systemd-journald deletes older journal records and journal files to avoid the use of a considerable amount of disk space, which is caused by the large amount of logs that are created. In some cases, the default rotate settings might not suffice your logging polices. If you need to control the disk space and how systemd deletes the logs, use the useful parameters that are listed in Table 8-2.

Table 8-2 systemd configuration settings

Option(s)	Description
SystemMaxUse= and RuntimeMaxUse=	Control how much disk space the journal can use up at maximum. Default to 10% of the respective file system size.
SystemKeepFree= and RuntimeKeepFree=	Control how much disk space systemd-journald shall always leaves free for other uses. Default is 15% of the respective file system size. The systemd-journald respects both limits, but uses the smaller value.

For more information, see the Linux manual by using the **`man 5 journald.conf`** command.

Use the following command to show the space on disk that is occupied by the journal data:

```
# journalctl --disk-usage
Archived and active journals take up 8.0M in the file system.
```

Table 8-3 lists the vacuum options that are available to help you to free up disk space that is used by journal.

Table 8-3 journalctl vacuum options

Option	Description
--vacuum-size=BYTES	Reduce disk usage by using the specified size.
--vacuum-files=INT	Leave only the specified number of journal files.
--vacuum-time=TIME	Remove journal files that are older than the specified time.

To reduce the log size from any value to 100M, run **`journalctl --vacuum-size=100M`** command.

To verify if everything is in order, issue the **`journalctl --verify`** command:

```
# journalctl --verify
PASS: /run/log/journal/95742106573a4215ac86da4a36c81e40/system.journal
```

Disabling and stopping the traditional log daemon (optional)

This step is optional. If you want, you can use the systemd-journald and the traditional log daemon. If you do not rely on the traditional log daemon, you can turn it off by using the following command:

```
# systemctl disable rsyslog.service
```

```
Removed /etc/systemd/system/syslog.service.
Removed /etc/systemd/system/multi-user.target.wants/rsyslog.service.
# systemctl stop rsyslog.service
```

8.3.2 Viewing the journal

Use the `journalctl` utility to view and filter the journal. If the `journalctl` utility is run without any parameters, it shows the entire journal in a format that is similar to traditional system log (syslog) files. The `journalctl` utility pipes the output into `less`, which is the default pager:

```
# journalctl
-- Logs begin at Wed 2020-09-16 13:37:16 EDT, end at Sat 2020-09-19 10:01:01 EDT.
--
Sep 16 13:37:16 rdbkrh04.pbm.ihost.com kernel: Linux version 4.18.0-193.el8.s390x
(mockbuild@s390-018.build.eng.bos.red>
Sep 16 13:37:16 rdbkrh04.pbm.ihost.com kernel: setup: Linux is running as a z/VM
guest operating system in 64-bit mode
Sep 16 13:37:16 rdbkrh04.pbm.ihost.com kernel: setup: The maximum memory size is
16384MB
...
```

Use the following command to view the journal and request the `less` pager to jump to the end of the log:

```
# journalctl -e
```

Use the following command to show the last twenty log events (which is similar to the **`tail -n20`** command):

```
# journalctl -n20
```

A live view of the journal can be obtained by using the **`--follow`** option. This option shows the last 10 lines (similar to the **`tail -f`** command) and then, displays updates when new log messages are added to the journal:

```
# journalctl -f
```

The live view can be stopped by pressing Ctrl+C to send a SIGINT signal to the journalctl process that causes the live view to end.

The default log output is intended to be legible. The following commands provide other output formats that are intended to be machine readable:

```
# journalctl -o verbose
# journalctl -o json-pretty
```

8.3.3 Filtering the journal

You can filter the large collections of data to help you view what you want to view. Use the commands in this section.

Use the following command to show the log messages of the current boot (filters out the messages from previous boots):

```
# journalctl -b
```

Use the following command to show today’s log messages:

```
# journalctl --since today
```

Use the following command to show kernel messages of the current boot only:

```
# journalctl -b -k
```

Use the following command to show only errors:

```
# journalctl -p err
```

Use the following command to show the log messages of a specific unit (such as the `sshd.service` unit):

```
# journalctl -u sshd.service
```

Use the following command to show the messages that are logged by a specific executable:

```
# journalctl /usr/sbin/sshd
```

Use the following command to show the messages that are logged by a specific PID, user ID, or group ID:

```
# journalctl _PID=0
# journalctl _UID=0
# journalctl _GID=0
```

For more fields, enter the **`man systemd.journal-fields`** command on the Linux command line.

8.4 System boot process

This section provides an overview of the boot process on Linux instances with systemd.

8.4.1 Boot loader

The zipl boot loader loads the Linux kernel with the initial RAM disk image (initramfs) and the boot parameters into memory and starts the kernel. Then, the kernel unpacks the initramfs and starts `/sbin/init`, which is provided by the initramfs (which is usually a symlink to systemd).

8.4.2 Initial RAM disk image

The initial RAM disk image (initramfs) contains the user space that is supposed to bring the disk that contains the root file system online and switch over to it. For example, if your rootfs is on a logical volume, it needs to activate the volume group. After the rootfs is mounted to /sysroot, it will be used as the new root directory. The initrd-switch-root.service executes `systemctl switch-root /sysroot` that switches the root directory and starts the systemd as PID 1. The job of the initramfs is done.

8.4.3 Systemd

Systemd is a system and service manager for Linux operating systems. From now on, systemd is responsible to start your system. Its goal is to reach the default target. First, it needs to load all of the unit files. Many of the unit files are generated dynamically by generators. For example, the systemd-fstab-generator creates mount units for the entries that are in the `/etc/fstab`. After systemd loads the unit files, it knows the dependency tree and activates all units that are necessary for the default target.

8.5 Analyzing Linux instances that use systemd

This section shows a few commands to retrieve performance statistics and information about the dependencies between systemd units.

8.5.1 Retrieving performance statistics

Use the following command to show the time that is spent in the various stages of the boot process after the boot loader starts the kernel:

```
# systemd-analyze time
Startup finished in 1.293s (kernel) + 5.031s (initrd) + 24.757s (userspace) =
31.082s
graphical.target was never reached
```

Use the following command to show the time that it took to initialize the service units, sorted by time:

```
# systemd-analyze blame
         30.327s pmlogger_daily.service
         11.279s pmcd.service
          8.798s pmlogger.service
          5.267s pmie.service
          2.674s NetworkManager-wait-online.service
          2.277s pmlogger_check.service
          1.805s tuned.service
          1.755s dracut-initqueue.service
          1.686s dracut-cmdline.service
           ...
```

Use the following command to display the time-critical chain of a systemd unit:

```
# systemd-analyze critical-chain default.target
The time after the unit is active or started is printed after the "@" character.
The time the unit takes to start is printed after the "+" character.

··multi-user.target @24.745s
  ··pmie.service @15.736s +5.267s
    ··pmcd.service @4.456s +11.279s
      ··network-online.target @4.455s
        ··NetworkManager-wait-online.service @1.780s +2.674s
          ··NetworkManager.service @1.715s +61ms
            ··network-pre.target @1.715s
            ...
```

Use the following command to generate a Scalable Vector Graphics (SVG) image that is similar to the sample image that is shown in Figure 8-1, which shows details about the boot process.

```
# systemd-analyze plot > plot.svg
```

Figure 8-1 systemd-analyze plot

Note: SVG files can be viewed easily by using your Internet browser.

8.5.2 Retrieving information about unit dependencies

With systemd, the following types of dependencies exist between units that affect:

- The activation of units (Requires/Wants/Conflicts)
- The order of units (After/Before)

Activation perspective (Requires/Wants/Conflicts)

Use the following command to show units that the specified unit requires or wants:

```
# systemctl list-dependencies sshd.service
? ··system.slice
? ··sshd-keygen.target
? · ··sshd-keygen@ecdsa.service
```

```
? · ··sshd-keygen@ed25519.service
? · ··sshd-keygen@rsa.service
? ··sysinit.target
  ...
```

Use the following command to show units that require or want the specified unit:

```
# systemctl list-dependencies --reverse sshd.service
sshd.service
? ··multi-user.target
?   ··graphical.target
```

Order perspective (After/Before)

Use the following command to list the units on which the specified unit has an “After” dependency. This command shows units that must be started before the specified unit.

```
# systemctl list-dependencies --after sshd.service
sshd.service
? ··system.slice
? ··systemd-journald.socket
? ··basic.target
? · ··-.mount
? · ··systemd-ask-password-plymouth.path
? · ··tmp.mount
...
```

Use the following command to list the units on which the specified unit has a “Before” dependency. This command shows units that need to be delayed until the specified unit starts.

```
# systemctl list-dependencies --before sshd.service
sshd.service
? ··multi-user.target
? · ··systemd-update-utmp-runlevel.service
? · ··graphical.target
? · · ··systemd-update-utmp-runlevel.service
? · · ··shutdown.target
? · ··shutdown.target
? ··shutdown.target

...
```

9

Working with containers

This chapter discusses the rise of containers and why they became a key component in the journey to the hybrid cloud.

As customers expectations continue to rise, customers became increasingly savvy and expect a highly personalized experience. Agility becomes more important than ever when trying to meet these expectations from a business perspective.

This chapter includes the following topics:

9.1 Container overview

Containers bring a new perspective to application developers when comparing them to a more traditional model that uses virtual machines. Containers can reduce the overhead on computing resources, which enables far more workloads to run on a physical server because each container shares a single host operating system.

A *container* is a unit of software that provides a packaging mechanism that creates an abstraction of the code and its dependencies to build faster and more reliable applications.

Unlike virtual machines that virtualize at the hardware level, containers virtualize at the operating system level. This feature makes containers lightweight and provides many advantages, such as a low hardware footprint, environment isolation, quick deployment, run everywhere, and reusability.

Figure 9-1 shows an architectural diagram in which applications that are running on virtual machines are compared to applications that are running on containers.

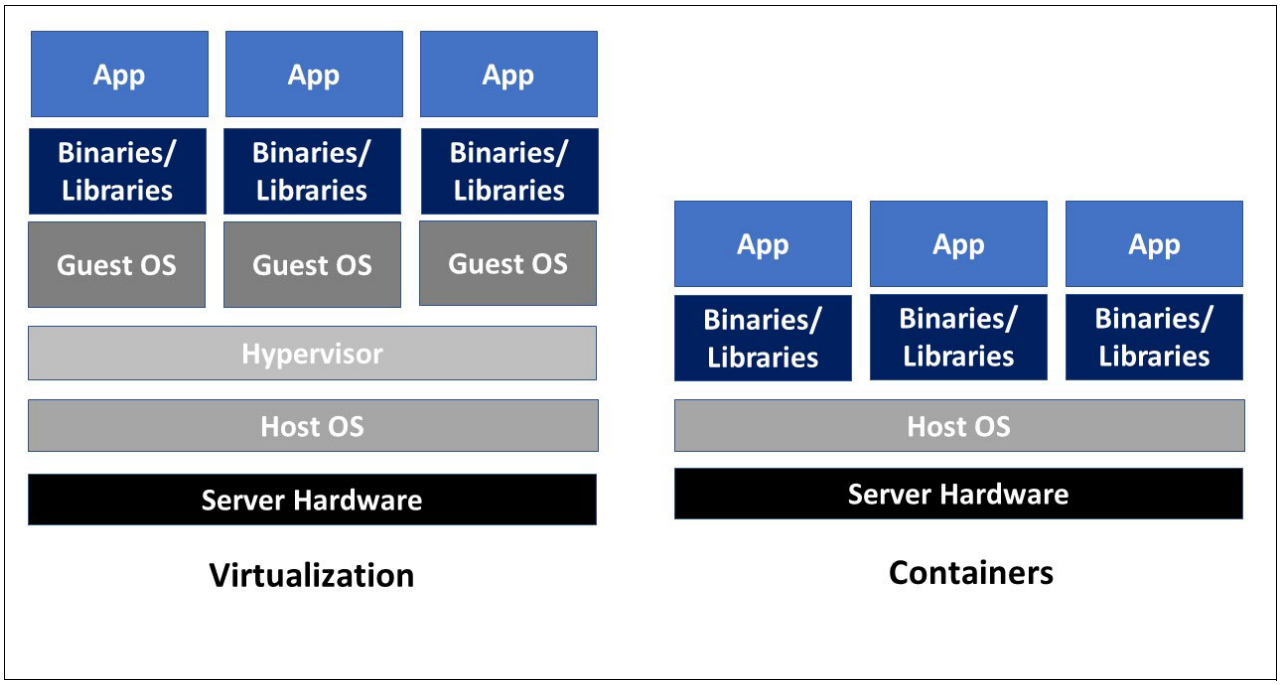

Figure 9-1 Comparing virtualization and containers

Containers often use a modular approach to deploy large and complex solutions. Instead of developing applications as single monolithic entities, containers help in designing applications that can be broken down into smaller components, each running in a separate container. This architecture is called *microservices*.

Containers also are portable and can be migrated between systems quickly and easily, especially when combined with Kubernetes.

Exploring this integration between containers and Kubernetes is out of the scope of this book. We explore only the container tooling that is included within Red Hat Enterprise Linux 8. For more information about containers, Kubernetes, and many other components, that make up a container's native platform, see this web page.

Red Hat removed the Docker container engine along with the **`docker`** command from Red Hat Enterprise Linux 8 entirely. For Red Hat Enterprise Linux 8, Docker is not included and not supported by Red Hat.

Red Hat Enterprise Linux 8 introduces a new set of container tools that allow users to find, run, build, and share containers. Podman, Skopeo, and Buildah are included with Red Hat Enterprise Linux 8. They cover different use cases: from simple to advanced to getting started with containers.

Figure 9-2 shows an architecture example of Docker tools and run time (on the left) and the new model that was introduced by Red Hat (on the right) by using podman, buildah, and skopeo and the podman container run time.

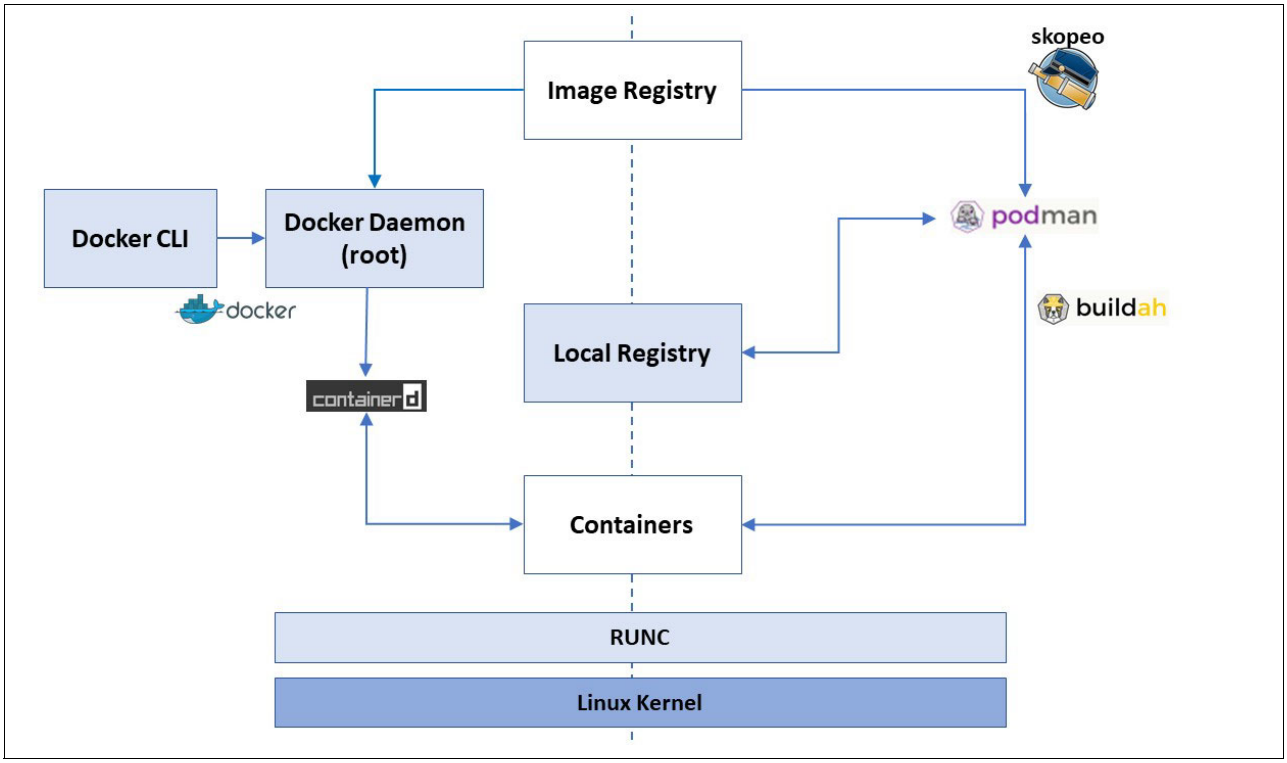

Figure 9-2 Docker tools and daemons versus Podman runtime and new container tools

In the enterprise, the focus is not on running individual containers from the command line. The primary venue for running containers is a Kubernetes-based platform, such as OpenShift.

All the new container tools that are available on Red Hat Enterprise Linux 8 are compliant with the Open Containers Initiative (OCI) standards, just like Docker, which allows you to build once and run anywhere.

Although the process to build a container image is the same across multiple architectures, if you build a container image for x86, it works on x86 only, which also is true for s390x and ppc64le. The portability of code is still a true statement if the image is binary-compatible with the host operating system.

9.2 Podman

Podman is a daemon-less container engine for developing, managing, and running OCI containers on Red Hat Enterprise Linux 8. Customers can build container images by using Podman and then, run it on OpenShift (which uses CRI-O) or other third-party, OCI-compliant container engines, and vice versa.

Podman is a container engine that works seamlessly with containers and pods (groups of containers that are deployed together on the same host). Podman also makes it easy to find, run, build, share, and deploy applications.

Because Podman is daemon-less, it starts containers and pods as child processes. Podman is the CLI tool for interacting with libpod, which is a library that allows other tools to manage pods and containers. Anyone that used the Docker CLI before sees that Podman offers a similar user syntax and user experience.

Installing Podman is a simple process. Run the command that is shown in Example 9-1.

Example 9-1 Installing Podman

```
# yum install podman

Updating Subscription Management repositories.
This system is registered to Red Hat Subscription Management, but is not receiving
updates. You can use subscription-manager to assign subscriptions.
Last metadata expiration check: 1:32:16 ago on Wed 23 Sep 2020 07:35:51 AM EDT.
Dependencies resolved.
================================================================================
 Package            Architecture   Version
Repository                        Size
================================================================================
Installing:
 podman             s390x         1.6.4-10.module+el8.2.0+6063+e761893a
InstallMedia-AppStream           11 M
Installing dependencies:
 conmon             s390x         2:2.0.6-1.module+el8.2.0+5182+3136e5d4
InstallMedia-AppStream           38 k
 libvarlink          s390x          18-3.el8
InstallMedia-BaseOS              41 k

Transaction Summary
================================================================================
Install  3 Packages

Total size: 11 M
Installed size: 55 M
Is this ok [y/N]:
```

Another way of installing Podman is to install the container tools group, which also includes all the container tools on Red Hat Enterprise Linux 8, as shown in Example 9-2

Example 9-2 Installing the container tools group

```
# yum install @container-tools

Updating Subscription Management repositories.
This system is registered to Red Hat Subscription Management, but is not receiving
updates. You can use subscription-manager to assign subscriptions.
Last metadata expiration check: 1:42:26 ago on Wed 23 Sep 2020 07:35:51 AM EDT.
Dependencies resolved.
================================================================================
 Package              Arch      Version
Repository                  Size
================================================================================
Installing group/module packages:
 cockpit-podman       noarch    12-1.module+el8.2.0+5950+6d183a6a
InstallMedia-AppStream     1.0 M
 conmon              s390x     2:2.0.6-1.module+el8.2.0+5182+3136e5d4
InstallMedia-AppStream      38 k
 podman              s390x     1.6.4-10.module+el8.2.0+6063+e761893a
InstallMedia-AppStream      11 M
 python-podman-api      noarch
1.2.0-0.2.gitd0a45fe.module+el8.2.0+5201+6b31f0d9      InstallMedia-AppStream
43 k
 skopeo              s390x     1:0.1.40-10.module+el8.2.0+5955+6cd70ceb
InstallMedia-AppStream     5.5 M
 toolbox             noarch    0.0.7-1.module+el8.2.0+6096+9c3f08f3
InstallMedia-AppStream      16 k
 udica               noarch    0.2.1-2.module+el8.2.0+4896+8f613c81
InstallMedia-AppStream      48 k
Installing dependencies:
libvarlink           s390x     18-3.el8
InstallMedia-BaseOS         41 k
 ostree-libs           s390x     2019.6-2.el8
InstallMedia-AppStream     343 k
 python3-pip           noarch    9.0.3-16.el8
InstallMedia-AppStream      20 k
 python3-psutil        s390x     5.6.3-5.el8
epel                                    395 k
 python3-setuptools     noarch     39.2.0-5.el8
InstallMedia-BaseOS           163 k
python36       s390x    3.6.8-2.module+el8.1.0+3334+5cb623d7   InstallMedia-AppStream
19 k
Installing module profiles:
 container-tools/common

Enabling module streams:
 python36                3.6


Transaction Summary
================================================================================
Install  13 Packages
```

```
Total size: 19 M
Total download size: 395 k
Installed size: 84 M
Is this ok [y/N]:
```

A trick that is used by Docker CLI veterans is to create an alias between Podman and Docker to facilitate the migration process to Podman, as shown in Example 9-3.

Example 9-3 Creating an alias to Podman

```
alias docker=podman
```

Podman can be used to pull images from container registries. However, to understand where Podman is pulling images from, run the command that is shown in Example 9-4.

Example 9-4 Default registries

```
# podman info

...
registries:
  blocked: null
  insecure: null
  search:
  - registry.access.redhat.com
  - registry.redhat.io
...
```

By default, Podman uses `registry.redhat.io`. To use this repository, you must have a valid subscription and your system that is suitably entitled, as described in Chapter 3, “Working with subscription-manager, yum, and DaNdiFied” on page 45.

Note: To pull container images from the Red Hat Container catalog by using Podman, Podman first must authenticate against that registry. To perform this authentication, use the following command:

```
# podman login -u <user_name>  https://registry.redhat.io
Password:
Login Succeeded!
```

The same process can be done for other registries that requires authentication:

```
# podman login -u <user_name> <remote_registry>
```

To pull container images from other registries (for example, `docker.io`), use the full image name as shown in Example 9-5.

Example 9-5 Pulling images with Podman

```
# podman pull docker.io/nginx
Trying to pull docker.io/nginx...
Getting image source signatures
Copying blob e08d2e78f348 done
Copying blob d059fd65bc2e done
Copying blob 569c48ee8402 done
```

```
Copying blob f86e35c583de done
Copying blob 07e4a6dbced6 done
Copying config 62c92e0066 done
Writing manifest to image destination
Storing signatures
62c92e00669ad27d552fab5e1947d2862eea3f04bbcdf0a4ee434202291059b0
```

Note: According to the OCI standards, the full image name is as follows:

`[registry]/[image]:[tag]`

For example:

`docker.io/nginx:latest`

It is also possible to configure extra registries on `/etc/containers/registries.conf`.

To verify the container images that were pulled to your system, use the **`podman images`** command, as shown in Example 9-6.

Example 9-6 Podman pulled images

```
# podman images
REPOSITORY                TAG      IMAGE ID       CREATED       SIZE
docker.io/library/nginx   latest   62c92e00669a   13 days ago   136 MB
```

A container image for the application NGINX is stored in this system. To run a container from this container image, use the **`podman run`** command, as shown in Example 9-7.

Example 9-7 Running a container from a container image

```
# podman run -dt --name our_nginx nginx
3654c9bc46642263666e4bace86ece54499227cc5e1d3a30ea15ec2c39d8ea7c
```

To verify that the container is running, use the **`podman ps`** command, as shown in Example 9-8.

Example 9-8 Verifying running containers

```
# podman ps
CONTAINER ID  IMAGE                            COMMAND               CREATED
STATUS            PORTS  NAMES
3654c9bc4664  docker.io/library/nginx:latest  nginx -g daemon o...  4 minutes ago
Up 4 minutes ago         our_nginx
```

To access this container, Podman offers the capability to inspect information about the running container instance (see Example 9-9).

Example 9-9 Inspecting containers

```
# podman inspect -f "{{.NetworkSettings.IPAddress}}" our_nginx
```

“10.88.0.2”

To verify NGINX is running inside that container instance, use the `curl` command, as shown by Example 9-10.

Example 9-10 Verifying running application inside the container

```
# curl 10.88.0.2
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

To stop that instance of NGINX, use the **`podman stop ID`** command. The ID can be found as a result of the command, as shown in Example 9-11.

Example 9-11 Using Podman to stop running containers

```
# podman stop 3654c9bc4664
3654c9bc46642263666e4bace86ece54499227cc5e1d3a30ea15ec2c39d8ea7c
```

To remove the stopped container use the **`rm`** command, as shown in Example 9-12.

Example 9-12 Using Podman to remove specific containers

```
# podman rm 3654c9bc4664
3654c9bc46642263666e4bace86ece54499227cc5e1d3a30ea15ec2c39d8ea7c
```

Only the container instance was removed by using the previous commands. The original container image is still available on the system as the **`podman images`** command reveals.

To completely remove the container image from the stem, use the sequence of commands that is shown in Example 9-13.

Example 9-13 Using Podman to query container images and remove them

```
# podman images
REPOSITORY                TAG      IMAGE ID       CREATED       SIZE
docker.io/library/nginx   latest   62c92e00669a   13 days ago   136 MB

# podman rmi 62c92e00669a
Untagged: docker.io/library/nginx:latest
Deleted: 62c92e00669ad27d552fab5e1947d2862eea3f04bbcdf0a4ee434202291059b0
```

By default, Podman uses the Red Hat Container Catalog to pull an image. Not all containers provide support for multi-architectures. It is recommended that you check the Red Hat Container Catalog frequently at this web page.

It is possible to extend the configuration for the default registries that Podman accesses. The default registries can be changed by editing the `/etc/containers/registries.conf` file, by commenting out the registries entry, `[registries.search]`, and encountering the registries, as shown Example 9-14.

Example 9-14 Modify default search registries

```
[registries.search]
#registries = ['registry.access.redhat.com', 'registry.redhat.io']
# The following registries entry can be used for convenience but includes
# container images built by the community. This set of content comes with all
# of the risks of any user generated content including security and performance
# issues. To use this list first comment out the default list, then uncomment
# the following list
registries = ['registry.access.redhat.com', 'registry.redhat.io', 'docker.io',
'quay.io']
```

This change allows other registries to be searched and used, as shown in Example 9-15.

Example 9-15 Podman pulling a container image

```
# podman pull open-liberty
Trying to pull registry.access.redhat.com/open-liberty...
  unsupported: This repo requires terms acceptance and is only available on
registry.redhat.io
Trying to pull registry.redhat.io/open-liberty...
  unknown: Not Found
Trying to pull docker.io/library/open-liberty...
Getting image source signatures
Copying blob 4164a02312fb done
Copying blob c68677136804 done
Copying blob 25294e13856f done
Copying blob a172b17642c4 done
Copying blob 201a00e3942c done
Copying blob 6ae1478bc3d6 done
Copying blob 64583a4ab9e3 done
Copying blob 04e4b14541a6 done
Copying blob d45cf93896c2 done
Copying blob 33be73490cc2 done
Copying blob 42eebcf35b51 done
```

```
Copying blob fe5dfcc6ff77 done
Copying blob 3bd83354e316 done
Copying blob 42fef5d53295 done
Copying config 8cc5185100 done
Writing manifest to image destination
Storing signatures
8cc5185100d75c44e7886293e6ea94179e3c1f4941960974b708b03d76e25245
```

Podman is a powerful tool that is used to pull and run containers, and build container images.

To build a container image, we used the instructions from this web page as a reference. However, we adapted them for use in a s390x container image for `noje.js` from `docker.io`.

Complete the following steps:

1. Create a directory to store all the files that are part of this build, as shown by Example 9-16.

Example 9-16 Creating a directory

```
mkdir sample_container_build
```

2. Create a file that is named `package.json` with the content that is shown in Example 9-17.

Example 9-17 Package.json file

```
{
  "name": "docker_web_app",
  "version": "1.0.0",
  "description": "Node.js on Docker",
  "author": "First Last <fmiranda@example.com>",
  "main": "server.js",
  "scripts": {
    "start": "node server.js"
  },
  "dependencies": {
    "express": "^4.16.1"
  }
}
```

3. Create a file a file that is named `server.js` with the content that is shown in Example 9-18.

Example 9-18 Server.js file

```
'use strict';

const express = require('express');

// Constants
const PORT = 8080;
const HOST = '0.0.0.0';

// App
const app = express();
app.get('/', (req, res) => {
  res.send('Hello World');
});
```

```
app.listen(PORT, HOST);
console.log(`Running on http://${HOST}:${PORT}`);
```

4. Create a file that is named `Dockerfile` with the content that is shown in Example 9-19.

Example 9-19 Dockerfile file

```
FROM s390x/node:latest

# Create app directory
WORKDIR /usr/src/app

# Install app dependencies
# A wildcard is used to ensure both package.json AND package-lock.json are copied
# where available (npm@5+)
COPY package*.json ./

RUN npm install
# If you are building your code for production
# RUN npm ci --only=production

# Bundle app source
COPY . .

EXPOSE 8080
CMD [ "node", "server.js" ]
```

5. Create a file that is named `dockerignore` with the content that is shown in Example 9-20.

Example 9-20 .dockerignore file

```
node_modules
npm-debug.log
```

6. Run the **podman** command to build the container, as shown in Example 9-21.

Example 9-21 Podman build command

```
# podman build -t fmiranda/node-web-app .
STEP 1: FROM s390x/node:latest
Getting image source signatures
Copying blob cc8faf537ff8 done
Copying blob 6abb293160bb done
Copying blob 13a7344ca59d done
Copying blob 9816eeae17f4 done
Copying blob faf135d701f8 done
Copying blob e6a933209363 done
Copying blob 80741dcaea5e done
Copying blob e9082185059e done
Copying blob 6a3670435671 done
Copying config 4807fadadf done
Writing manifest to image destination
Storing signatures
STEP 2: WORKDIR /usr/src/app
409bff8efa3beafb8420d5b50d1edaa72a961d54cf07a4b6effa3e30df42176a
STEP 3: COPY package*.json ./
a66ec974309dca44e124ce355b7919140f95ffdd507b801d2b3d16607135fd08
```

```
STEP 4: RUN npm install
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN docker_web_app@1.0.0 No repository field.
npm WARN docker_web_app@1.0.0 No license field.

added 50 packages from 37 contributors and audited 50 packages in 1.218s
found 0 vulnerabilities

0bc4c0a9548d31e08b70630bd868d5bc33b5f88cf9bf765f449c1d3dcff63263
STEP 5: COPY . .
119e9dfa459af84229e5b25cad42a39f779d34c38d6284a647736c9a3c2730f1
STEP 6: EXPOSE 8080
ab65df39bf6a0951f90df793d1f340ae14a307ac8f4b0ad8b8eb43e772da9cd9
STEP 7: CMD [ "node", "server.js" ]
STEP 8: COMMIT fmiranda/node-web-app
62b9fd7635130669451a2eef9626dac8e735dde541a841b4cd60f79be1028054
```

7. Verify the created container image by using the **`podman images`** command. Run the container image that specifies what ports to use and then, check whether the container is running, as shown in Example 9-22.

Example 9-22 Podman commands

```
# podman images
REPOSITORY                          TAG      IMAGE ID       CREATED         SIZE
localhost/fmiranda/node-web-app     latest   62b9fd763513   9 seconds ago   972 MB
docker.io/s390x/node                latest   4807fadadffa   2 months ago    968 MB
docker.io/s390x/alpine              latest   0d9ffb5b0ee7   3 months ago    5.88 MB


# podman run -p 49160:8080 -d fmiranda/node-web-app
6c9c91b7ded322063c8c68a37ab975bc4b2271a62c7b89d1522897a3e72f84dc

# podman ps
CONTAINER ID  IMAGE                                    COMMAND        CREATED
STATUS             PORTS                     NAMES
6c9c91b7ded3  localhost/fmiranda/node-web-app:latest  node server.js  38 seconds
ago  Up 37 seconds ago  0.0.0.0:49160->8080/tcp  peaceful_joliot
```

8. Test the running container, as shown in Example 9-23.

Example 9-23 Testing running container

```
# curl -i localhost:49160
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 11
ETag: W/"b-Ck1VqNd45QIvq3AZd8XYQLvEhtA"
Date: Fri, 25 Sep 2020 13:34:34 GMT
Connection: keep-alive

Hello World
```

The Podman examples that are shown in this section demonstrate only a fraction of Podman's capabilities. For more information about Podman, see this web page.

For more examples, see this web page.

Podman is one of the container tools that is included with Red Hat Enterprise Linux 8.

9.3 Buildah

Podman specializes in all of the commands and functions that help you to maintain and modify OCI container images, such as pulling and tagging. It also helps to create, run, and maintain those containers.

Although Podman uses Buildah's build function under the covers to create a container image, the two projects features some differences. The major difference between Podman and Buildah is their *concept* of a container.

Podman creates traditional containers. The intent of these containers is to be controlled through the entirety of a container lifecycle (pause, checkpoint, restore, and so on). Buildah containers are created to allow content to be added to the container image.

Buildah's commands replicate all of the commands that are found in a Dockerfile. It helps with container builds without requiring a Dockerfile. Buildah also allows the use of other scripting languages to build container images without requiring a daemon.

Buildah is an efficient way to create OCI images; Podman allows you to manage and maintain those images and containers in a production environment by using familiar container CLI commands. Together, they form a strong foundation to support your OCI container image and container needs.

To install the Buildah container tool, use the command that is shown in Example 9-24.

Example 9-24 Installing Buildah

```
# yum install buildah
Updating Subscription Management repositories.
Last metadata expiration check: 0:32:12 ago on Sat 26 Sep 2020 10:16:39 AM EDT.
Dependencies resolved.
================================================================================
 Package             Architecture         Version
Repository                                              Size
================================================================================
Installing:
 buildah                     s390x
1.14.9-1.module+el8.2.1+6689+748e6520
rhel-8-for-s390x-appstream-rpms                     8.5 M

Transaction Summary
================================================================================
Install  1 Package

Total download size: 8.5 M
Installed size: 37 M
Is this ok [y/N]:
Y
```

To get started with Buildah, complete the following steps:

1. To create the container image by using the Buildah commands, create a file that is named `buildah_helloapp.sh` with the content that is shown Example 9-25. The command tat is shown in Example 9-25 creates a container image that is based on Fedora 32 for s390x. Then, it compiles a hello world application.

Example 9-25 buildah_helloapp.sh file

```
#!/usr/bin/env bash

set -o errexit

# Create a container
container=$(buildah from fedora:latest)

# Labels are part of the "buildah config" command
buildah config --label maintainer="Filipe Miranda <fmiranda@ibm.com>" $container

# Grab the source code outside of the container
curl -sSL http://ftpmirror.gnu.org/hello/hello-2.10.tar.gz -o hello-2.10.tar.gz

buildah copy $container hello-2.10.tar.gz /tmp/hello-2.10.tar.gz

buildah run $container dnf install -y tar gzip gcc make
buildah run $container dnf clean all
buildah run $container tar xvzf /tmp/hello-2.10.tar.gz -C /opt

# Workingdir is also a "buildah config" command
buildah config --workingdir /opt/hello-2.10 $container

buildah run $container ./configure
buildah run $container make
buildah run $container make install
buildah run $container hello -v

# Entrypoint, too, is a "buildah config" command
buildah config --entrypoint /usr/local/bin/hello $container

# Finally saves the running container to an image
buildah commit --format docker $container hello:latest
```

2. Run the **`buildah_helloapp.sh`** script, as shown in Example 9-26. The process to build this container generates a lengthy output. Example 9-26 shows a cropped output of the **`buildah_helloapp.sh`** script when it runs.

Example 9-26 Running buildah_helloapp.sh

```
# ./buildah_helloapp.sh
Getting image source signatures
Copying blob 258eddf3cf51 done
Copying config da1e9e9be4 done
Writing manifest to image destination
Storing signatures
f040c0f962d5b594cdafc36b12e5937830fcd23d722210373e7d64bc89496098
```

```
Fedora 32 openh264 (From Cisco) - s390x
5.3 kB/s | 2.5 kB     00:00
Fedora Modular 32 - s390x
8.7 MB/s | 4.9 MB     00:00
Fedora Modular 32 - s390x - Updates
5.8 MB/s | 3.7 MB     00:00
Fedora 32 - s390x - Updates
16 MB/s |  19 MB     00:01
Fedora 32 - s390x
28 MB/s |  63 MB     00:02
...

Getting image source signatures
Copying blob 8eefc7c7dae2 skipped: already exists
Copying blob 802382454284 done
Copying config ae3b8bc5e7 done
Writing manifest to image destination
Storing signatures
ae3b8bc5e70044329a15b9efc117e8bf664f20e793291c1fda04ef7e3a0dd898
```

3. Verify that the container image was created by using the **podman images** command, as shown in Example 9-27.

Example 9-27 Podman images to show available container images

```
# podman images
REPOSITORY                TAG      IMAGE ID       CREATED         SIZE
localhost/hello           latest   ae3b8bc5e700   4 minutes ago   359 MB
docker.io/library/fedora  latest   da1e9e9be4ad   2 months ago    193 MB
```

4. Run the container image by specifying the binary that was created for the hello world application inside the container. Also, verify whether the message Hello World is printed as the output (see Example 9-28).

Example 9-28 Testing the newly created binary inside the container image

```
# podman run -it hello:latest /usr/local/bin/hello

Hello, world!
```

Buildah can also be used to create containers (as with Podman) that are based on Dockerfiles. For convenience, Podman is more likely to be used to create containers that are based on Dockerfiles because Buildah is a more advanced tool that provides many more advanced features when building a container image.

Example 9-29 shows a simple Dockerfile that creates a container image that is based on a Red Hat container image from Red Hat Enterprise Linux 8 from the Red Hat Container Catalog.

The following steps show you how to access the Red Hat registry and then create the necessary files to build the final container image:

1. Create the Dockerfile and the script that prints a hello world message when the container runs, as shown in Example 9-29.

Example 9-29 Dockerfile

```
FROM registry.redhat.io/ubi8/ubi
ADD hello_world /usr/local/bin
ENTRYPOINT "/usr/local/bin/hello_world"
```

2. Create the `hello_world` script, as shown in Example 9-30.

Example 9-30 hello_world script file

```
echo "Hello World!!"
```

3. Build the container image by using the **`buildah`** command, as shown in Example 9-31.

Example 9-31 Using buildah to build a Dockerfile

```
# buildah bud -t hello_world .
STEP 1: FROM registry.redhat.io/ubi8/ubi
Getting image source signatures
Copying blob 34fd41f98O1e done
Copying blob O24e289f9d3O done
Copying config b2a8209190 done
Writing manifest to image destination
Storing signatures
STEP 2: ADD hello_world /usr/local/bin
STEP 3: ENTRYPOINT "/usr/local/bin/hello_world"
STEP 4: COMMIT hello_world
Getting image source signatures
Copying blob a2d86e78aee8 skipped: already exists
Copying blob a3cd5b7d5e9f skipped: already exists
Copying blob 227abf8bbef6 done
Copying config ef0cf68367 done
Writing manifest to image destination
Storing signatures
ef0cf68367fb3a63d81f9df443c236987abddbf92e76b506bfcf19211105951e
ef0cf68367fb3a63d81f9df443c236987abddbf92e76b506bfcf19211105951e
```

4. Verify the container images, as shown in Example 9-32.

Example 9-32 Newly created container images

```
# podman images
REPOSITORY                    TAG      IMAGE ID       CREATED         SIZE
localhost/hello_world         latest   ef0cf68367fb   7 minutes ago   208 MB
registry.redhat.io/ubi8/ubi   latest   b2a820919039   3 weeks ago     208 MB
```

5. Run the container image, as shown in Example 9-33.

Example 9-33 Running the hello_world container

```
# podman run localhost/hello_world
Hello World!!
```

In this section, we outlined and demonstrated only a few of the many features of Buildah. Also, with Buildah, you can perform the following tasks:

- Build a container from a Dockerfile.
- Mount a container's root file system to add or change content (**buildah mount**).
- Use the updated contents of a container's root file system as a file system layer to commit content to a new image (**buildah commit**).
- Unmount a mounted container (**buildah umount**).
- Remove a container (**buildah rm**) or a container image (**buildah rmi**).

9.4 Skopeo

Although Buildah builds container images and Podman runs container images, Skopeo transfers container images. Skopeo works with images in remote repositories and provides for the transfer, inspection, and deletion of images.

Registries can include the Docker registry, your own local registries, Red Hat Quay, or OpenShift registries.

To install Skopeo run the command that is shown in Example 9-34.

Example 9-34 Installing Skopeo

```
# yum install skopeo
Updating Subscription Management repositories.
Last metadata expiration check: 0:36:55 ago on Sat 26 Sep 2020 10:16:39 AM EDT.
Dependencies resolved.
================================================================================
 Package                 Architecture       Version
Repository                                        Size
================================================================================
Installing:
 skopeo                                s390x
1:1.0.0-1.module+el8.2.1+6676+604e1b26
rhel-8-for-s390x-appstream-rpms                  6.3 M
Upgrading:
 containers-common                     s390x
1:1.0.0-1.module+el8.2.1+6676+604e1b26
rhel-8-for-s390x-appstream-rpms                   52 k

Transaction Summary
================================================================================
Install  1 Package
Upgrade  1 Package

Total download size: 6.4 M
Is this ok [y/N]:
Y
```

To inspect a container image from a registry, you must identify the container format, location of the registry, and repository or image.

Note: To inspect a container from a remote registry, use the full name of that image:

`registry_address/repository/image:tag`

Example 9-35 shows an example of how the **skopeo** command can be used to inspect a remote container image.

Example 9-35 Using Skopeo

```
# skopeo inspect docker://docker.io/open-liberty

{
    "Name": "docker.io/library/open-liberty",
    "Digest":
"sha256:e6d7315adc4eafb7d9557b1e84fd9fcf1c6d308807639e6873692afcea056f43",
    "RepoTags": [
        "18.0.0.4-javaee7-java8-ibm",
        "18.0.0.4-javaee7-java8-ibmsfj",
        "18.0.0.4-javaee7-java8-openj9",
        "18.0.0.4-javaee7",
        "18.0.0.4-javaee8-java8-ibm",
        "18.0.0.4-javaee8-java8-ibmsfj",
        "18.0.0.4-javaee8-java8-openj9",
        "18.0.0.4-javaee8",
        "18.0.0.4-kernel-java8-ibm",
        "18.0.0.4-kernel-java8-ibmsfj",
        "18.0.0.4-kernel-java8-openj9",
        "18.0.0.4-kernel",
...
"Architecture": "s390x",
    "Os": "linux",
...
```

The **skopeo inspect** command that is shown in Example 9-35 generates a long list of information about that container image (the architecture type points to s390x). Open Liberty is a container image that provides multi-architecture support; that is, multiple container images of Open Liberty are available on `Docker.io` and because we inspected from an s390x system, it returns the relevant information to the architecture where the query originated.

The output is presented in a JSON format. To get output that is better formatted from the **skopeo inspect** command, install **qt** (a command line JSON processor) by using the command that is shown in Example 9-36.

Example 9-36 Installing a JSON processor

```
# yum install jq
Updating Subscription Management repositories.
Last metadata expiration check: 3:02:23 ago on Sun 27 Sep 2020 09:06:57 AM EDT.
Dependencies resolved.
================================================================================
 Package                    Architecture              Version
Repository                                          Size
================================================================================
Installing:
```

```
 jq                                      s390x
1.5-12.el8                             InstallMedia-AppStream
165 k
Installing dependencies:
 oniguruma                               s390x
6.8.2-1.el8                            InstallMedia-AppStream
184 k

Transaction Summary
================================================================================
Install  2 Packages

Total size: 349 k
Installed size: 1.1 M
Is this ok [y/N]: y
```

Run the command that is shown in Example 9-37 to get a more polished output.

Example 9-37 Inspecting container images and treating them with the json processor

```
# skopeo inspect docker://docker.io/open-liberty | jq

"Created": "2020-08-28T19:43:09.617866101Z",
  "DockerVersion": "18.09.7",
  "Labels": {
    "org.opencontainers.image.authors": "Arthur De Magalhaes, Chris Potter",
    "org.opencontainers.image.description": "This image contains the Open Liberty
runtime with AdoptOpenJDK with OpenJ9 and Ubuntu as the base OS.  For more
information on this image please see
https://github.com/OpenLiberty/ci.docker#building-an-application-image",
    "org.opencontainers.image.revision": "cl200920200820-0913",
    "org.opencontainers.image.source": "https://github.com/OpenLiberty/ci.docker",
    "org.opencontainers.image.title": "Open Liberty",
    "org.opencontainers.image.url": "https://openliberty.io/",
    "org.opencontainers.image.vendor": "Open Liberty",
    "org.opencontainers.image.version": "20.0.0.9"
  },
  "Architecture": "s390x",
  "Os": "linux",
....
```

To determine whether the container image supports other architectures types, use the command that is shown in Example 9-38.

Example 9-38 Skopeo command to query all supported architectures for that container image

```
skopeo inspect --raw  docker://docker.io/open-liberty | jq
{
  "manifests": [
    {
      "digest":
"sha256:704ba328eb98a4eef7d793d962f135db72ede61b5c7cfb94f67e1b607d590035",
      "mediaType": "application/vnd.docker.distribution.manifest.v2+json",
      "platform": {
        "architecture": "amd64",
        "os": "linux"
```

```
      },
      "size": 3250
    },
    {
      "digest":
"sha256:e62cf575816e65abc6d28111e14bc41869d7dfa8d4baaad61877d3ee27e789b1",
      "mediaType": "application/vnd.docker.distribution.manifest.v2+json",
      "platform": {
        "architecture": "ppc64le",
        "os": "linux"
      },
      "size": 3250
    },
    {
      "digest":
"sha256:ed7e9a4b446478fb7f4e8939d401762a1d8f2bee92fd014fafd36686daf604e8",
      "mediaType": "application/vnd.docker.distribution.manifest.v2+json",
      "platform": {
        "architecture": "s390x",
        "os": "linux"
      },
      "size": 3250
    }
  ],
  "mediaType": "application/vnd.docker.distribution.manifest.list.v2+json",
  "schemaVersion": 2
}
```


Skopeo is command line utility that performs various operations on container images and image registries. Skopeo can perform the following tasks:

- Copy an image from and to various storage mechanisms.
- Inspect a remote image that shows its properties including its layers, without requiring you to pull the image to the host.
- Delete an image from an image repository.
- When required by the repository, pass the suitable credentials and certificates for authentication.

This chapter covered only a few of Skopeo's capabilities. For more information, see Red Hat's official Skopeo documentation.

10

Security features

Security is an important cornerstone in many ways and became the most crucial topic for enterprise executives. Security is not only about the most valuable assets many companies have (their data), but also about avoiding data breaches. The risk of data breaches is high for smaller companies and large enterprises, ISPs, partners, vendors, and even global multi-national companies.

Security in information technology is a broad field that includes the following aspects:

- Authentication to ensure identity (Single Sign On, certificates)
- Key Exchange to exchange cryptographic keys and perform handshaking for confidentiality and non-repudiation
- Confidentiality to ensure that a message can be read only by the intended receiver (encryption) to allow intended recipient
- Integrity to ensure that a received message is still the original message and was not altered (hash/MAC) during the transit or any attacks in the network
- Non-repudiation to ensure that a message really came from a certain sender (signature) and make sure it is not from anyone else.

Every organization that handles customer information, such as personally identifiable information (PII), or offers services through the internet by using the web (on-premises or in the cloud) must ensure that processed data is secured against all kind of threats and attacks.

All security controls and measures must be taken to prevent any data leakage and to ensure system and data integrity. It is no longer sufficient to state that data processing is secure; rather, it is a continuous process. You must able to provide suitable verification and compliance reports to auditors on-demand, and comply with regulations to establish trust in your services. Most importantly, you must prevent a loss of revenue and reputation that can result from the lack of proper security measures.

Therefore, it is a recommended practice to establish the strongest security procedures and processes at all levels of data processing, including the physical security of the machine rooms at your data center (controlling access to the facilities) and implementing suitable access levels to applications, programs, data, archives, and so on. The principle of least privilege must be implemented at all levels based on Role Based Access Controls (RBAC).

This chapter provides an overview of the Red Hat Enterprise Linux 8.2 security features along with the IBM Z hardware cryptographic modules to achieve end-to-end encryption and security controls that are available to achieve system and data integrity goals.

In this chapter, the following topics are included:

10.1 Red Hat Enterprise Linux 8.2: Security hardening

Security begins even before you start installing Red Hat Enterprise Linux. Configuring your system securely from the beginning of the process makes it easier to implement more security settings later.

Password protection for the BIOS (or BIOS equivalent) and the boot loader can prevent unauthorized users who have physical access to systems from booting by using removable media or obtaining root privileges through single user mode.

It is a good practice to create separate partitions for the `/boot`, `/`, `/home`, `/tmp`, and `/var/tmp/` directories. For example, the `/boot` partition is the first partition that is read by the system during start. The boot loader and kernel images that are used to start your system into Red Hat Enterprise Linux 8 are stored in this partition. During the installation process, you can encrypt partitions.

It is always a best practice to restrict the network to only known traffic and install only the minimum set of packages that are required for a specific environment. After the installation, the update command is used to apply any security-related procedures, as shown in Example 10-1.

Example 10-1 Applying updates

```
[root@rdbkrh07 ~]# yum -y update
Updating Subscription Management repositories.
Red Hat  Enterprise Linux 8 - BaseOS2.7 MB/s | 2.7 kB     00:00
Red Hat  Enterprise Linux 8 - AppStream3.1 MB/s | 3.2 kB     00:00
Dependencies resolved.
Nothing to do.
Complete!
[root@rdbkrh07 ~]#
```

Typically, the firewall service, `firewalld`, is automatically enabled with the installation of Red Hat Enterprise Linux. However, some scenarios exist in which it might be disabled; re-enabling in the kickstart configuration, as described in Chapter 2, “Automated Red Hat Enterprise Linux installations by using Kickstart” on page 35. In such a case, it is recommended to consider re-enabling the firewall.

The firewall service can be checked, as shown in Figure 10-1.

```
[root@rdbkrh07 ~]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor p
   Active: active (running) since Thu 2020-09-24 07:01:39 EDT; 8h ago
     Docs: man:firewalld(1)
 Main PID: 1125 (firewalld)
    Tasks: 2 (limit: 101854)
   Memory: 31.3M
   CGroup: /system.slice/firewalld.service
           └─1125 /usr/libexec/platform-python -s /usr/sbin/firewalld --nofork

Sep 24 07:01:38 rdbkrh07.pbm.ihost.com systemd[1]: Starting firewalld - dynamic
Sep 24 07:01:39 rdbkrh07.pbm.ihost.com systemd[1]: Started firewalld - dynamic
Sep 24 07:01:39 rdbkrh07.pbm.ihost.com firewalld[1125]: WARNING: AllowZoneDrift
```

Figure 10-1 firewalld service

It is always a good security practice to disable services that might not be required. For example, a printer service, such as the ‘cups’ service, can be disabled by using the **systemctl disable cups** command.

A vulnerability assessment is an internal audit of your network and system security; the results of which indicate the confidentiality, integrity, and availability of your network. Vulnerability assessments can be broken down into one of two types: outside looking in and inside looking around. Various tools are available for vulnerability assessment, including the following sample tools:

- **Nmap**

  This tool can be used to find host systems and open ports on those systems. To install **nmap** from the AppStream repository, enter the **yum install nmap** command as the root user.

  For example, by using the **-o** (the letter O) option along with the **nmap** command, you can enable operating system detection and reveal operating system information for the hosts, as shown in Example 10-2.

  Example 10-2 Using nmap

```
[root@rdbkrh07 ~]# nmap -O 129.40.23.138
Starting Nmap 7.70 ( https://nmap.org ) at 2020-09-25 00:10 EDT
Nmap scan report for rdbkrh07.pbm.ihost.com (129.40.23.138)
Host is up (0.0000040s latency).
Not shown: 995 closed ports
PORT  STATESERVICE
21/tcpopenftp
22/tcpopenssh
5801/tcpopenvnc-http-1
5901/tcpopenvnc-1
9090/tcpopenzeus-admin
Device ype: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.7 - 3.10
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at
https://nmap.org/
submit/ .
Nmap done: 1 IP address (1 host up) scanned in 4.19 seconds
[root@rdbkrh07 ~]#
```

- **oscap**

  This command-line utility and the SCAP Workbench graphical utility (**scap-workbench**) provide a fully automated compliance audit by allowing a user to scan their local systems, validate security compliance content, and generate reports and guides that are based on these scans and evaluations.

- **AIDE**

  Advanced Intrusion Detection Environment (AIDE) is a utility that creates a database of files on the system. It then uses that database to ensure file integrity and detect system intrusions[1]

[1] https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/security_hardening/checking-integrity-with-aide_security-hardening

10.2 SELinux security policies

Crypto policies is a system component that configures the core cryptographic subsystems, which cover the TLS, IPsec, SSH, DNSSec, and Kerberos protocols[2]. It provides a small set of policies that the administrator can select.

After a system-wide policy is set up, applications in Red Hat Enterprise Linux follow it and refuse to use algorithms and protocols that do not meet the policy, unless you specifically request the application to do so.

Red Hat Enterprise Linux 8 supports four policies: DEFAULT, FIPS, FUTURE, and LEGACY, as listed in Table 10-1.

Table 10-1 System-wide cryptographic policy levels

Name	Description
DEFAULT	Offers secure settings for current threat models. It allows the TLS 1.2, 1.3, IKEv2, and SSH2 protocols. The RSA keys and Diffie-Hellman parameters are accepted if they are at least 2048 bits long.
LEGACY	Maximum compatibility with Red Hat Enterprise Linux 5 and earlier, is less secure because of an increased attack surface. In addition to the DEFAULT algorithms and protocols, it includes support for the TLS 1.0 and 1.1 protocols. The algorithms DAs, 3DES, and RC4 are allowed; RSA keys and Diffie-Hellman parameters are accepted if they are at least 1023 bits long.
FUTURE	A more conservative security level. This level does not allow the use of SHA-1 in signature algorithms. The RSA keys and Diffie-Hellman parameters are accepted if they are 3072 bits long.
FIPS	Conforms with the Federal Information Processing Standards (FIPS)140-2 requirements. Used internally by the **`fips-mode-setup`** tool, which switches the Red Hat Enterprise Linux system into FIPS mode. To check the current mode, use the following command: `[root@rdbkrh07 ~]# fips-mode-setup --check` For more information, see this web page.

To view or change the system-wide cryptographic policy, use the `update-crypto-policies` application, as shown in Example 10-3.

Example 10-3 Set and query the system's policy

```
[root@rdbkrh07 ~]# update-crypto-policies --show
DEFAULT
[root@rdbkrh07 ~]#
[root@rdbkrh07 ~]# update-crypto-policies --set FUTURE
Setting system policy to FUTURE
[root@rdbkrh07 ~]#
```

10.2.1 SELinux

Security Enhanced Linux (SELinux) provides another layer of system security. SELinux fundamentally answers the question: May <subject> do <action> to <object>?; for example: May a web server access files in users' home directories?

[2] https://access.redhat.com/articles/3666211

The standard access policy that is based on the user, group, and other permissions, is known as Discretionary Access Control (DAC). It does not enable system administrators to create comprehensive and fine-grained security policies.[3]

SELinux implements Mandatory Access Control (MAC). Every process and system resource includes a special security label that is called an SELinuxcontext. ASELinuxcontext, sometimes referred to as an SELinux label, is an identifier that abstracts away the system-level details and focuses on the security properties of the entity.

The SELinux policy uses these contexts in a series of rules that define how processes can interact with each other and the various system resources. By default, the policy does not allow any interaction unless a rule specifically grants access.

SELinux contexts features several fields: user, role, type, and security level. The SELinux type information is perhaps the most important when it comes to the SELinux policy as the most common policy rule. SELinux types end with `_t`.

For example, in the type name for the web server `httpd_t`, the type context for files and directories normally found in `/var/www/html/` is `httpd_sys_content_t`. A policy rule is used that permits Apache (the web server process that is running as `httpd_t`) to access files and directories with a context that is found often in `/var/www/html/` and other web server directories, such as `httpd_sys_content_t`.

For more information about SELinux, see this web page.

SELinux on IBM Z

For more information about SELinux for Linux on IBM Z, see *Security for Linux on System z*, S24-7728.

10.3 Red Hat Enterprise Linux 8.2 System Roles

Red Hat Enterprise Linux System Roles is a collection of Ansible roles and modules. Red Hat Enterprise Linux System Roles provide a configuration interface to remotely manage multiple Red Hat Enterprise Linux systems. The interface enables managing system configurations across multiple versions of Red Hat Enterprise Linux, and adopting new major releases.

On Red Hat Enterprise Linux 8, the interface consists of the following roles:

- kdump
- network
- selinux
- storage
- timesync

All these roles are provided by the rhel-system-roles package that is available in the AppStream repository.

10.3.1 Creating Red Hat Enterprise Linux 8 system roles

The `rhel-system-roles` package MUST be installed on the host that acts as a control node.

[3] https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/pdf/using_selinux/red_hat_enterprise_linux-8-using_selinux-en-us.pdf

As a part of the `rhel-system-roles`, Ansible must be running as a control node on the system.

Note: Ansible Control Node is not supported on IBM Z, but Ansible can be run on an x86 machine to control the system roles on IBM Z.

10.3.2 Red Hat Enterprise Linux 8.2: Insights

Red Hat Insights, which is included as part of a Red Hat Enterprise Linux subscription at no extra cost, can help with operational efficiency use cases in the areas of performance, stability, and availability, and security and compliance use cases. Insights can also evaluate your Red Hat Enterprise Linux installation for a list of Common Vulnerabilities and Exposures (CVEs) that affect your systems, or evaluate compliance issues, such as PCI or DISA-STIG.

Insights can show you applicable errata or patches, identify differences in installed packages between systems or as compared to a baseline. It even allows you to create custom policies for criteria, such as alerting you when a firewall is turned off.

At a high level, all you must do is install the Insights client and register.

For each Red Hat Enterprise Linux host, use SSH to log on to the host system with an account that has sudo permissions and then, run the following commands:

```
# yum install insights-client

# insights-client --register
```

Example 10-4 shows typical output from the install and register commands.

Example 10-4 Installing and registering the Insights client

```
[root@rdbkrh07 ~]# yum install insights-client
Updating Subscription Management repositories.
RedHat Enterprise Linux 8 - BaseOS           2.7 MB/s | 2.7 kB     00:00
RedHat Enterprise Linux 8 - AppStream        3.1 MB/s | 3.2 kB     00:00
Package insights-client-3.0.13-1.e18_1.noarch is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@rdbkrh07 ~]#
[root@rdbkrh07 ~]# insights-client --register
You successfully registered 2551f28c-377d-49b3-a6dc-db4078e01298 to account
6252206.
Successfully registered host rdbkrh07.pbm.ihost.com
Automatic scheduling for Insights has been enabled.
Starting to collect Insights data for rdbkrh07.pbm.ihost.com
Uploading Insights data.
Successfully uploaded report from rdbkrh07.pbm.ihost.com to acount 6252206.
View the Red Hat Insights console at https://cloud.redhat.com/insights/
[root@rdbkrh07 ~]#
```

After the `insights-client` is installed, you can log on to `https://cloud.redhat.com/insights/` to browse to the Insights Dashboard, as shown in Figure 10-2.

Some of the services now offered by Red Hat Insights include:

- Advisor identifies known configuration risks in the operating system, underlying infrastructure, or workloads that affect performance, stability, availability, or security best practices.
- Vulnerability assesses, remediates, and reports on CVEs that affect Red Hat Enterprise Linux environments in the cloud or on-premises.
- Compliance analyzes the level of compliance of a Red Hat Enterprise Linux environment to an OpenSCAP policy, based on the Red Hat Security guide.
- Patch determines which Red Hat product advisories apply to an organization's specific Red Hat Enterprise Linux instances. also It provides guidance for manual remediation manually or by way of Ansible Playbooks for patching.
- Drift compares systems to baselines, system histories, and to each other to troubleshoot or identify differences.
- Policies enable organizations to define and monitor for policies that are important internally, with alerts for environments that are not aligned to a policy.

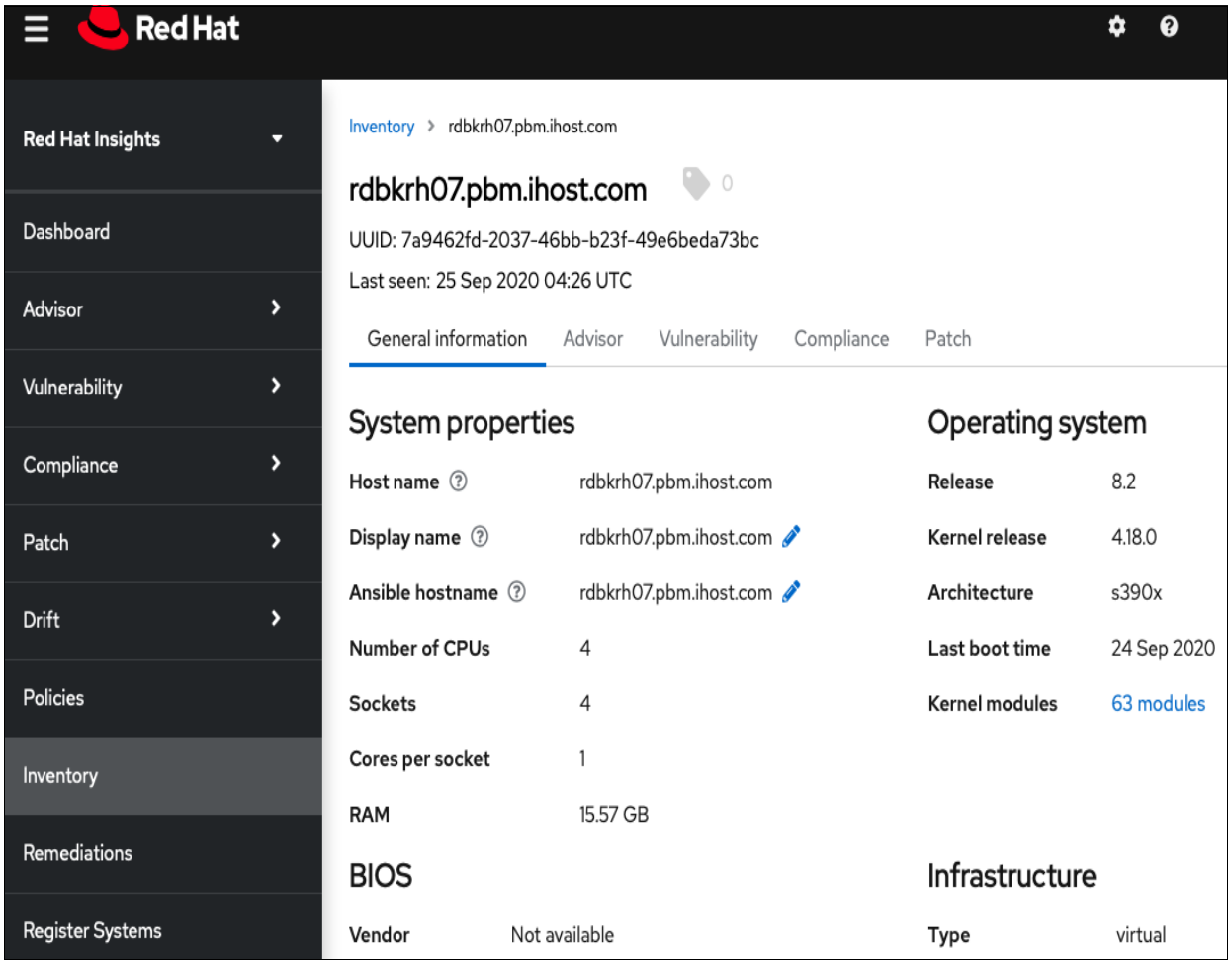

Figure 10-2 Insights Dashboard

10.4 Hardware cryptographic support

Security and data protection are essential business imperatives and auditing and regulatory compliance is increasing in complexity when building modern hybrid solutions. The use of encryption is one of the best ways to reduce the risks and financial losses of a data breach and meet complex compliance mandate. However, implementing encryption can be a complex process for many organizations.

Enterprises must consider the following points:

- Who is accessing the data?
- What systems and applications are affected?
- What data must be encrypted (PII, PCI-DSS, and so on)
- Where must encryption occur? Must consider end-to-end encryption to avoid any type of third-party attacks.
- Who is responsible for encryption? Roles and privileges must be defined and a least access policy used.

Data is a major asset. Encryption policies must cover data in-transit and data at-rest, but not require costly development efforts to attain this objective. Organizations need a transparent and reusable approach to enable extensive encryption of data in-flight and at-rest to achieve compliance and security protocols.

With solutions around privileged identity management, sensitive data protection, and integrated security intelligence, IBM Z security offers the next generation of secure, trusted transactions along with Red Hat Enterprise Linux 8 security features, such as SELinux and system roles.

10.4.1 Pervasive encryption with IBM Z

Pervasive encryption is a data-centric approach to information security that entails protecting data that is entering and exiting the IBM Z platform. It involves encrypting data in-transit and data at-rest. It is a paradigm shift from selective encryption (where only the data that is required to achieve compliance is encrypted) to pervasive encryption. Pervasive encryption with IBM Z is enabled through tight platform integration that includes the following features:

- Integrated cryptographic hardware: Central Processor Assist for Cryptographic Function (CPACF) is a co-processor on every processor unit that accelerates encryption. Crypto Express features can be used as hardware security modules (HSMs).
- Data set and file encryption: You can protect Linux file systems and IBM z/OS data sets by using policy-controlled encryption that is transparent to applications and databases.
- Network encryption: You can protect network data traffic by using standards-based encryption from endpoint to endpoint.
- Full disk encryption: You can use disk drive encryption that protects data at-rest when disk drives are retired, sent for repair, or repurposed.
- CF encryption: Secures the parallel sysplex infrastructure, including the CF links and data that is stored in the CF by using policy-based encryption.

Advantages of hardware cryptography

Hardware cryptography features the following major advantages:

- Encryption

  IBM Z provides hardware encryption support that can be used to reduce the effect of expensive encryption operations.

- Performance

  Because the encryption operations are offloaded to the IBM Z CPACF processor or to the Crypto Express6S card, performance and the throughput of your workload are less affected.

  To verify CPACF, you must first install the package `libica` if it does not exist, as shown in Example 10-5.

Example 10-5 install package libica

```
[root@rdbkrh07 ~]# yum install -y libica
pdating Subscription Management repositories.
RedHat Enterprise Linux 8 - BaseOS           2.7 MB/s | 2.7 kB     00:00
RedHat Enterprise Linux 8 - AppStream        3.1 MB/s | 3.2 kB     00:00
Package insights-client-3.0.13-1.e18_1.noarch is already installed.
Dependencies resolved.
================================================================================
 Package   Arch     Version               Repository                    Size
================================================================================
Installing:
 libica    s390x    3.6.1-2.e18_2.1       rhel-8-for-s390x-baseos-rpms 128k

Transaction Summary
================================================================================
Install 1 Package
```

To use CPACF, you must also install the Licensed Internal Code (LIC) feature 3863 (Crypto Enablement feature), which is available at no extra charge. Install the Crypto Enablement (LIC 3863) feature, even if you do not intend to use the Crypto Express feature because it provides considerable benefits to an active CPACF.

You can check whether the CPACF is enabled on your environment by using the HMC by performing the following steps.

a. From the HMC, browse to your IBM Z system by clicking **Tasks Index** → **Single Object Operations** → **IBM Z System** → **System Details**.

Check for the ‘CP Assist for Crypto functions: Installed, as shown in Figure 10-3.

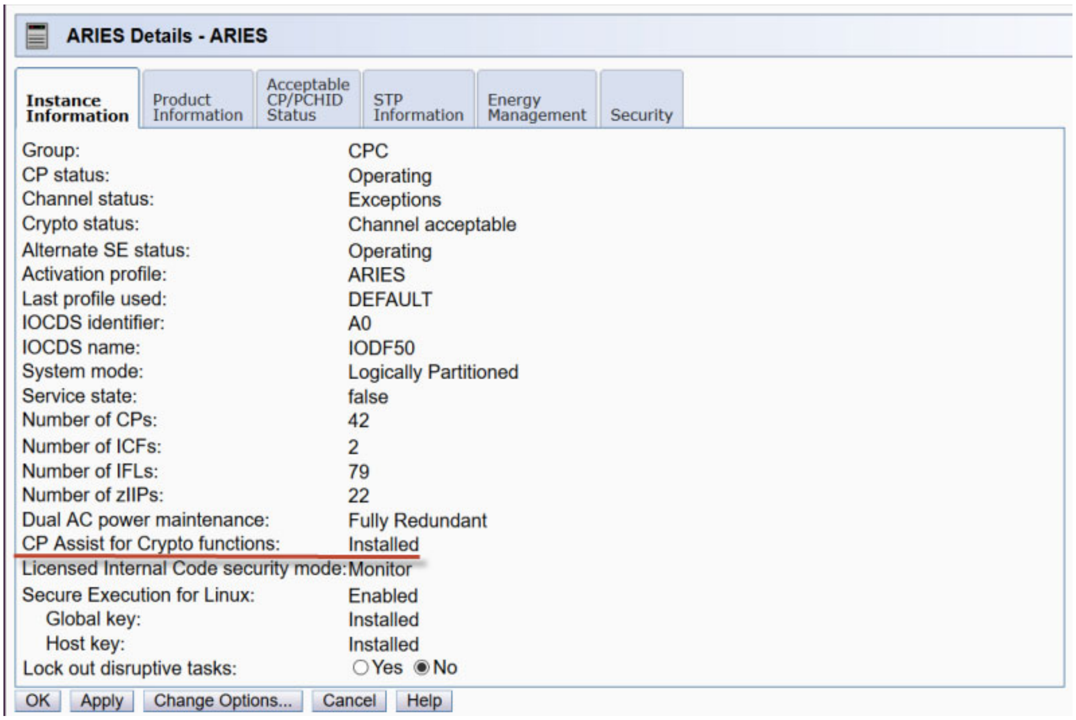

Figure 10-3 CP Assist for Crypto functions

b. From Red Hat Enterprise Linux 8, run the **`icainfo`** command to verify that CPACF is installed and which algorithms are supported in the hardware. The **`icainfo`** command provides an overview of the supported algorithms with modes of operations and how they are implemented on your Linux system (hardware, software, or both).

As shown in Example 10-6, available hardware support is presented in two columns: dynamic hardware and static hardware. Dynamic hardware with a `yes` in the hardware column indicates that hardware is supported by cryptographic coprocessors and static hardware indicates it is supported by CPACF. A `no`' in the software column indicates that no software fallback that is provided by OpenSSH is implemented in libica for this function.

Example 10-6 Command icainfo output

```
[root@rdbkrh07 ~]#  icainfo
              Cryptographic algorithm support
---------------------------------------------------------
                 |           hardware         |
  function       | dynamic  |   static        |  software
-----------------+----------+-----------------+----------
           SHA-1 |   no     |    yes          |    yes
         SHA-224 |   no     |    yes          |    yes
         SHA-256 |   no     |    yes          |    yes
         SHA-384 |   no     |    yes          |    yes
```

```
      SHA-512 |   no      |    yes         |     yes
      SHA-224 |   no      |    yes         |     yes
  SHA-512/224 |   no      |    yes         |     yes
  SHA-512/256 |   no      |    yes         |     yes
     SHA3-224 |   no      |    yes         |     no
     SHA3-256 |   no      |    yes         |     no
     SHA3-384 |   no      |    yes         |     no
     SHA3-512 |   no      |    yes         |     no
    SHAKE-128 |   no      |    yes         |     no
    SHAKE-256 |   no      |    yes         |     no
        GHASH |   no      |    yes         |     no
       P_RING |   no      |    yes         |     yes
DDRBG-SHA-512 |   no      |    yes         |     yes
         ECDH |   no      |    yes         |     no
```

If the Crypto Enablement Feature 3863 is installed, you see that other algorithms are available for hardware support, as shown in Example 10-7.

Example 10-7 Command icainfo output

```
      DES ECB |   no      |    yes         |     yes
      DES CBC |   no      |    yes         |     yes
      DES OFB |   no      |    yes         |     no
      DES CFB |   no      |    yes         |     no
      DES CTR |   no      |    yes         |     no
     DES CMAC |   no      |    yes         |     no
     3DES ECB |   no      |    yes         |     yes
     3DES CBC |   no      |    yes         |     yes
     3DES OFB |   no      |    yes         |     no
     3DES CFB |   no      |    yes         |     no
     3DES CTR |   no      |    yes         |     no
    3DES CMAC |   no      |    yes         |     no
      AES ECB |   no      |    yes         |     yes
      AES CBC |   no      |    yes         |     yes
      AES OFB |   no      |    yes         |     no
      AES CFB |   no      |    yes         |     no
      AES CTR |   no      |    yes         |     no
     AES CMAC |   no      |    yes         |     no
      AES XTR |   no      |    yes         |     no
      AES GCM |   no      |    yes         |     no

Built-in FIPS support: FIPS mode inactive.
[root@rdbkrh07 ~]#
```

If you find a `no` in the software column in the output of the **`icainfo`** command (see Figure 10-12), no software fallback is implemented in `libica`. Also, you can see that FIPs support is not active in the system. FIPs mode can be activated as discussed in 10.2, “SELinux security policies” on page 185.

- Key Management

  IBM Z key management system is designed to meet FIPS Level 4 standards.

10.4.2 Hardware Cryptography in RHEL on IBM Z

Linux on z/VM with access to a shared cryptographic accelerator can observe an accelerator or a coprocessor, but not both. Figure 10-4 shows a typical crypto card deployment.

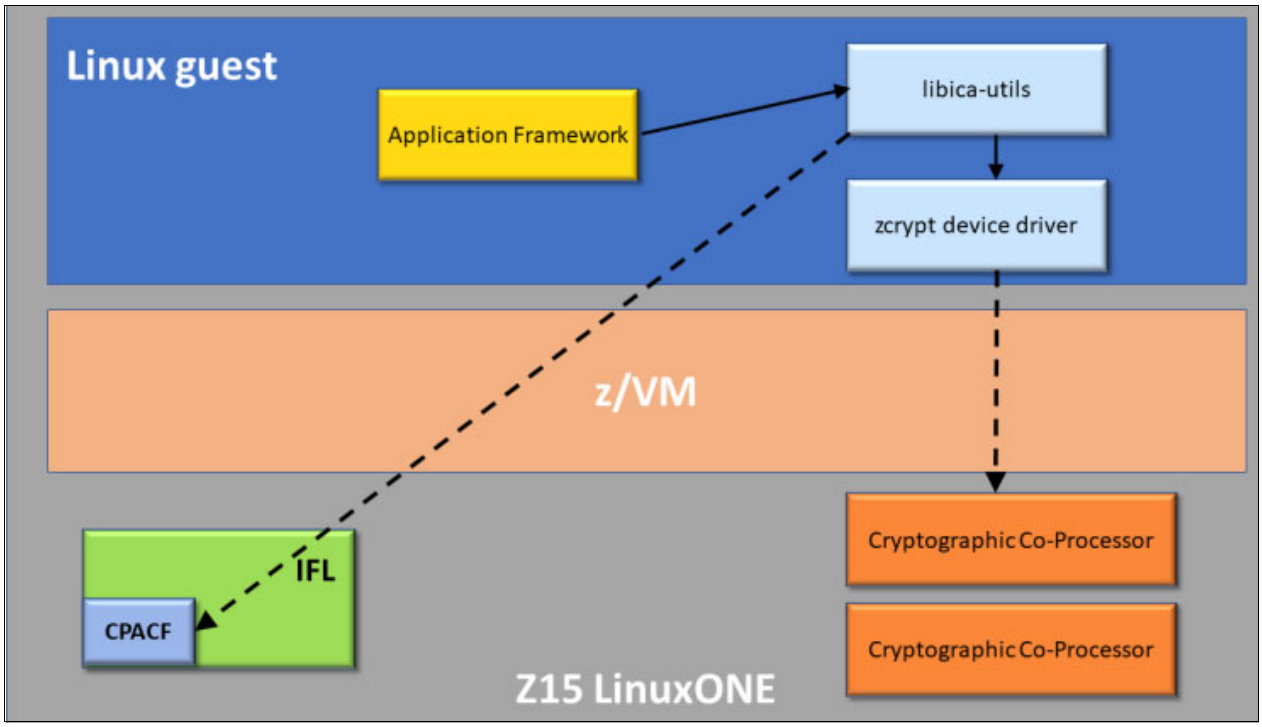

Figure 10-4 Linux on IBM Z device drivers and libraries

At the lowest level, the zcrypt device driver provides an API that often is not used directly by applications. Instead, this API is intended for intermediate software layers, which in turn provide more sophisticated cryptographic functions to the next upper level of code.

The libica Linux library is an intermediate library that offers various cryptographic functions. Some of these functions are performed by the hardware cryptographic devices that are under control of the device driver.

CPACF enablement verification

Any user of Red Hat Enterprise Linux on IBM Z can easily check whether the Crypto Enablement feature is installed and which algorithms are supported in hardware. Hardware-acceleration for DES, TDES, AES, and GHASH requires CPACF.

Review the output from the `cat /proc/cpuinfo` file to see whether the CPACF feature is enabled on your hardware. If the features list has `msa` listed, it means that CPACF is enabled, as shown in Example 10-8.

Example 10-8 CPACF verification

```
[root@rdbkrh07 ~]# cat /proc/cpuinfo
vendor_id       : IBM/S390
# processors    : 4
bogomips per cpu: 24038.00
max thread id   : 0
features        : esan3 zarch stfle msa ldisp eimm dfp edat etf3eh highgprs te v
x vxd vxe gs vxe2 vxp sort dflt sie
```

```
facilities      :  0 1 2 3 4 5 6 7 8 9 10 12 14 15 16 17 18 19 20 21 22 23 24 25
26
 27 ....
```

For the Linux virtual machine to gain access to the crypto card, you must load a specialized crypto device driver. You can check whether it is loaded by using the command **lszcrypt,** as shown in

Example 10-9 Check whether device drivers are loaded

```
[root@rdbkrh07 ~]# lszcrypt
CARD.DOMAIN  TYPE    MODE          STATUS     REQUESTS
-----------------------------------------------------
01           CEX7A   Accelerator    online           2
01.0001      CEX7A   Accelerator    online           2
[root@rdbkrh07 ~]#
```

Note: Ensure the s390-tools package is installed before the **lszcrypt** command is used. Usually, this package is installed by default. If it is not installed by default, install it manually.

10.4.3 Pervasive Encryption: Data at-rest

Using cryptographic capabilities of IBM Z, you can speed up the process of protecting file system data by using encryption. Linux provides encryption support for the following areas:

- In-kernel cryptography
- Cryptographic support for development programs or applications

In-kernel cryptography is used when the Linux kernel performs encryption requests by using in-kernel modules. Because the in-kernel modules are not available for user programs and applications, specific encryption libraries are available.

To use hardware support for encryption, the in-kernel cryptography modules or the cryptography libraries must be aware of the available hardware and use it. You can verify the supported hardware cryptographic operations by using the command that is shown in Example 10-10. The line that ends in -s390 indicates the hardware acceleration for a corresponding algorithm or mode.

Example 10-10 Check whether hardware acceleration exists

```
[root@rdbkrh07 ~]# cat /proc/crypto | grep driver
driver        : ghash-s390
driver        : gcm-aes-s390
driver        : ctr(aes-generic)
driver        : ctr-aes-s390
driver        : xts(ecb(aes-s390))
driver        : xts-aes-s390
driver        : cbc(aes-generic)
driver        : cbc-aes-s390
driver        : ecb(aes-s390)
driver        : ecb-aes-s390
driver        : aes-s390
driver        : ctr-des3_ede-s390
driver        : ctr-des3-s390
driver        : cbc-des3_ede-s390
```

```
driver       : ecb-des3_ede-s390
driver       : des3_ede-s390
driver       : cbc-des-s390
driver       : ecb-des-s390
driver       : des-s390
```

The kernel implements cryptographic operations for kernel subsystems, such as `dm-crypt` and IPsec. The `dm-crypt` subsystem in Linux is implemented as a device mapper that can be stacked on top of other devices that are managed through the device mapper framework. Therefore, you can encrypt from entire disks to software RAID volumes and LVM logical volumes, which adds flexibility to your encryption strategy.

The cryptographic operations that can be accelerated by hardware implementations depend on your IBM Z hardware features and mode of operating Red Hat Enterprise Linux 8.2 The administration of `dm-crypt` is done by using `cryptsetup`, which features Linux Unified Key Setup (LUKS). LUKS standardizes the format of the encrypted disk, which allows different implementations (even from other operating systems) to access and decrypt the disk.

Standard Linux kernels include modules that use the CPACF capabilities of the IBM Z hardware. These modules are optimized to use the inherent IBM Z hardware features to provide better performance and security.

The hardware-dependent modules for the CPACF often are in the `/lib/modules` directory unless they were compiled into the kernel. Example 10-11 shows the commands to use to ensure that the modules were included.

Example 10-11 Checking hardware-dependent modules

```
[root@rdbkrh07 crypto]#
[root@rdbkrh07 crypto]# pwd
/usr/lib/modules/4.18.0-193.el8.s390x/kernel/arch/s390/crypto
[root@rdbkrh07 crypto]# ls -la
total 48
drwxr-xr-x. 2 root root  194 Sep 21 16:54 .
drwxr-xr-x. 8 root root   84 Sep 21 16:54 ..
-rw-r--r--. 1 root root 7256 Mar 27 11:01 aes_s390.ko.xz
-rw-r--r--. 1 root root 4284 Mar 27 11:01 des_s390.ko.xz
-rw-r--r--. 1 root root 2428 Mar 27 11:01 ghash_s390.ko.xz
-rw-r--r--. 1 root root 4912 Mar 27 11:01 paes_s390.ko.xz
-rw-r--r--. 1 root root 7084 Mar 27 11:01 prng.ko.xz
-rw-r--r--. 1 root root 2320 Mar 27 11:01 sha3_256_s390.ko.xz
-rw-r--r--. 1 root root 2356 Mar 27 11:01 sha3_512_s390.ko.xz
-rw-r--r--. 1 root root 2524 Mar 27 11:01 sha512_s390.ko.xz
[root@rdbkrh07 crypto]#
```

If you want to ensure that the CPACF hardware is used for the encryption, load the IBM Z optimized `aes_s390` kernel module by using the **`modprobe`** command. Check it by using the **`lsmod`** command and then, list the modules by using the **`cat proc/crypto`** command, as shown in Example 10-12.

Example 10-12 Loading the aes_s390 module

```
[root@rdbkrh07 crypto]# modprobe aes_s390
[root@rdbkrh07 crypto]# lsmod | grep aes_s390
aes_s390          24576  0
[root@rdbkrh07 crypto]# cat /proc/crypto | grep aes-s390
driver       : gcm-aes-s390
driver       : ctr-aes-s390
driver       : xts(ecb(aes-s390))
```

```
driver       : xts-aes-s390
driver       : cbc-aes-s390
driver       : ecb(aes-s390)
driver       : ecb-aes-s390
driver       : aes-s390
```

The **cryptsetup** feature command provides an interface for configuring encryption on block devices (such as /home or swap partitions) by using the Linux kernel device mapper target dm-crypt. It features integrated LUKS support. LUKS standardizes the format of the encrypted disk, which allows different implementations (even from other operating systems) to access and decrypt the disk. Example 10-13 shows it is installed.

Example 10-13 Install package 'cryptsetup'

```
[root@rdbkrh07 crypto]#
[root@rdbkrh07 crypto]# yum install cryptsetup
Updating Subscription Management repositories.
RedHat Enterprise Linux 8 - BaseOS              2.7 MB/s | 2.7 kB     00:00
RedHat Enterprise Linux 8 - AppStream           3.1 MB/s | 3.2 kB     00:00
Package cryptsetup-2.2.2-1.el8.s390x is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@rdbkrh07 crypto]# _
```

The dm-crypt feature supports various cipher and hashing algorithms that you can select from those algorithms that are available in the Kernel and listed in the /proc/crypto procfs file. That is, dm-crypt takes advantage of the unique hardware acceleration features of IBM Z that increase encryption and decryption speed.

By using the **cryptsetup** command, a LUKS partition can be created on the respective disk devices. For full disk encryption, use the AES-xts hardware feature.

10.4.4 Pervasive Encryption: Data in-transit

This section describes how to use the cryptographic functions of the IBM Z to encrypt data in-transit (flight). This technique means that the data is encrypted and decrypted before and after it is transmitted. We use OpenSSL to demonstrate the encryption of data in-transit (flight).

To check whether OpenSSL is installed in Red Hat Enterprise Linux 8.2, run the command that is shown in Figure 10-5.

Figure 10-5 Checking whether OpenSSL is available

```
[root@rdbkrh07 ~]# rpm -qa | grep openssl
openssl-1.1.1c-15.el8.s390x
openssl-pkcs11-0.4.10-2.el8.s390x
openssl-libs-1.1.1c-15.el8.s390x
apr-util-openssl-1.6.1-6.el8.s390x
[root@rdbkrh07 ~]#
```

During the installation of Red Hat Enterprise Linux 8.2, the `openssl-ibmca` package was not automatically installed and needed to be installed manually, as shown in Figure .

```
[root@rdbkrh07 crypto]# yum install openssl-ibmca
Updating Subscription Management repositories.
RedHat Enterprise Linux 8 - BaseOS              2.7 MB/s | 2.7 kB     00:00
RedHat Enterprise Linux 8 - AppStream           3.1 MB/s | 3.2 kB     00:00
Dependencies resolved.
================================================================================
 Package            Arch       Version           Repository                Size
================================================================================
Installing:
 openssl-ibmca      s390x      2.1.0-1.el8       InstallMedia-BaseOS       41 k

Transaction Summary
================================================================================
Install  1 Package

Total size: 41 k
Installed size: 98 k
Is this ok [y/N]: y
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : openssl-ibmca-2.1.0-1.el8.s390x                        1/1
  Running scriptlet: openssl-ibmca-2.1.0-1.el8.s390x                        1/1
  Verifying        : openssl-ibmca-2.1.0-1.el8.s390x                        1/1
Installed products updated.

Installed:
  openssl-ibmca-2.1.0-1.el8.s390x

Complete!
[root@rdbkrh07 crypto]#
```

Figure 10-6 Installing Package openssl-ibmca

To use the `ibmca` engine and gain the benefits from cryptographic hardware support, you must modify the OpenSSL configuration file. Complete the following steps to configure this file:

1. Locate the OpenSSL configuration file, `tls`. In our lab environment, it is in the following subdirectory:

   `/etc/pki/tls`

2. Take a back-up copy of the configuration file by using the command that is shown in Figure 2.

```
[root@rdbkrh07 crypto]#
[root@rdbkrh07 crypto]# ls /etc/pki/tls
cert.pem  certs  ct_log_list.cnf  misc  openssl.cnf  private
[root@rdbkrh07 crypto]#
[root@rdbkrh07 crypto]#
[root@rdbkrh07 crypto]# ls -la /etc/pki/tls/openssl.cnf
-rw-r--r--. 1 root root 11225 Mar  5  2020 /etc/pki/tls/openssl.cnf
[root@rdbkrh07 crypto]#
[root@rdbkrh07 crypto]# cp -p /etc/pki/tls/openssl.cnf /etc/pki/tls/openssl.cnf.
backup
[root@rdbkrh07 crypto]# ls -al /etc/pki/tls/openssl.cnf*
-rw-r--r--. 1 root root 11225 Mar  5  2020 /etc/pki/tls/openssl.cnf
-rw-r--r--. 1 root root 11225 Mar  5  2020 /etc/pki/tls/openssl.cnf.backup
[root@rdbkrh07 crypto]# _
```

Figure 10-7 Backing up openssl.cnf

3. Some `ibmca`-related content must be appended to the `openssl.cnf` file. This code is included in the `openssl-ibmca` package. Locate the `ibmca` package (see Figure 3) and look for a file that is called `openssl.cnf.sample.s390x`.

```
[root@rdbkrh07 crypto]#
[root@rdbkrh07 crypto]# ls -l /usr/share/doc/openssl-ibmca/openssl.cnf.sample.s3
90x
-rw-r--r--. 1 root root 1771 Nov  6  2019 /usr/share/doc/openssl-ibmca/openssl.c
nf.sample.s390x
[root@rdbkrh07 crypto]#
```

Figure 10-8 Locating the ibmca file

4. Append the `ibmca-related` configuration lines to the OpenSSL configuration file by using the command that is shown in Figure 10-9.

```
[root@rdbkrh07 crypto]#
[root@rdbkrh07 crypto]# tee -a /etc/pki/tls/openssl.cnf < /usr/share/doc/openssl
-ibmca/openssl.cnf.sample.s390x
#
```

Figure 10-9 Appending the ibmca file to openssl.cnf

5. Verify the append to ensure that the `ibmca` file is appended to the `openssl.cnf` (see Figure 10-10).

```
[root@rdbkrh07 crypto]#
[root@rdbkrh07 crypto]# grep -n ibmca_section /etc/pki/tls/openssl.cnf
385:ibmca = ibmca_section
387:[ibmca_section]
[root@rdbkrh07 crypto]#
```

Figure 10-10 Verifying successful append

You also can log on to an SSH session from your workstation to verify the package.

For more information, see this web page.

11

Helpful information

This chapter contains miscellaneous helpful information. These topics help facilitate administration, save time, increase functions, or add capabilities to your systems.

This chapter includes the following topics:

11.1 Rescuing a Linux system

This section describes how to start your Linux server in different modes for troubleshooting purposes. It covers starting Linux in single user mode. It also describes how to enter a rescue environment when you require more advanced troubleshooting.

11.1.1 Initrd shell and systemd targets

Before Red Hat Enterprise Linux 7, SysV offered special runlevels to use for specific tasks, for example, the single user mode or the emergency mode. The **systemd** command introduces a new concept that is called *targets* and offers the same function in a different way. For more information about target units, see 8.2.2, “Managing systemd target units” on page 152.

Using initrd shell and systemd targets with Red Hat Enterprise Linux

In this section, we provide an example of the use of the initrd shell and systemd targets to enter single user mode or rescue target mode on a Fibre Channel Protocol (FCP) Linux virtual machine that is named LINUX2.

Note: Use the **SET LOADDEV** command to identify the location of a program to be loaded as a result of a guest IPL from SCSI disk. For more information, see this web page.

Table 11-1 lists the WWPN and LUN ID that is used in this example.

Table 11-1 SAN information

FCP channel	WWPN (Port) storage	LUN ID (disk)
B747	0x5005076309141145	0x4001402200000000

Complete the following steps:

1. IPL your Linux server from the 3270 console by using the **SET LOADDEV** command to prepare the SAN disk to be IPLed in the x3270 console. Specify the target port and LUN of the SCSI boot disk. Enter the command as shown in the following example:

   ```
   #cp set loaddev portname <wwpn> lun <lun> scpdata '<kernel_parameters>'
   ```

   Where:

   <wwpn> specifies the world wide port name (WWPN) of the target port in hexadecimal format. A blank separates the first eight digits from the final eight digits.

   <lun> specifies the LUN of the SCSI boot disk in hexadecimal format. A blank separating the first eight digits from the final eight digits. In our lab environment, the following command was used:

   ```
   ===> cp set loaddev portname 50050763 09141145 lun 40014022 00000000
   ```

2. Specify kernel parameters by using the following command

   ```
   #cp set loaddev scpdata <APPEND|NEW> '<kernel_parameters>'
   ```

   Where:

   <APPEND> appends kernel parameters to existing SCPDATA. This setting is the default and is optional.

   <NEW> replaces existing SCPDATA (optional).

<kernel_parameters> specifies a set of kernel parameters to be stored as system control program data (SCPDATA). When Linux is stated, these kernel parameters are concatenated to the end of the kernel parameters that are used by your start configuration. <kernel_parameters> must contain ASCII characters only. If characters other than ASCII characters are used, the start process ignores the SCPDATA.

In our example, we tell the Linux kernel to load in rescue mode by using the following command:

```
===> cp set loaddev scpdata APPEND 'systemd.unit=rescue.target'
```

3. Start the IPL and boot process by entering the following command:

   ```
   #cp i <devno>
   ```

   Where:

   <devno> is the device number of the FCP channel that provides access to the SCSI start disk. In our environment, Example 11-1 shows the command that we used and an excerpt of its output.

Note: If you need to find the FCP channel that is attached to the server, issue the following command:

```
cp query fcp
OO: FCP  B747 ON FCP   B747 CHPID 25 SUBCHANNEL = 0000
OO:      B747 DEVTYPE FCP         VIRTUAL CHPID 25 FCP REAL CHPID 25
OO:      B747 QDIO-ELIGIBLE       QIOASSIST-ELIGIBLE
OO:      B747 DATA ROUTER ELIGIBLE
OO:      WWPN C05076D08002C49C
```

Example 11-1 Starting the IPL

```
===> cp ipl b747
OO: HCPLDI2816I Acquiring the machine loader from the processor controller.
OO: HCPLDI2817I Load completed from the processor controller.
OO: HCPLDI2817I Now starting the machine loader.
....Output Omitted...
[    0.166854] Kernel command line: root=/dev/mapper/vg_system-root
crashkernel=auto rd.zfcp=0.0.b747,0x50050763091b1145,0x400140220
0000000 rd.zfcp=0.0.b747,0x5005076309141145,0x4001402200000000
rd.lvm.lv=vg_system/root cio_ignore=all,!condev rd.znet=qeth,0.0.0640
,0.0.0641,0.0.0642,layer2=1,portno=0,portname=Z systemd.unit=rescue.target
....Output Omitted...
You are in rescue mode. After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, "systemctl default" or "exit"
to boot into default mode.
Give root password for maintenance
(or press Control-D to continue):
```

   The single user mode requires the root password.

In rescue.target mode, all of the file systems in /etc/fstab are mounted, but networking is not started. To exit single user mode, enter exit to continue to start normally to the default systemd target in your system.

To enter a different systemd target, from the IPL command enter the target that you want in the **systemd.unit=** parameter. For example, to enter the `emergency.target`, we used LINUX2 in the commands that are shown in Example 11-2.

Example 11-2 Enter an alternative systemd target

```
===> cp set loaddev portname 50050763 09141145 lun 40014022 00000000
===> cp set loaddev scpdata NEW 'systemd.unit=emergency.target'
===> ipl b747
```

Note: Because we entered **system.unit=rescue.target** in the **set scpdata** command in Example 11-2, we specified **NEW** instead of **APPEND** in the following command to replace scpdata with the new content.

Optionally, you can use **CP QUERY** command to confirm **loaddev** parameters:

```
CP QUERY LOADDEV
PORTNAME 50050763 09141145    LUN  40014002 00000000    BOOTPROG 0
BR_LBA   00000000 00000000
SCPDATA
     0----+----1----+----2----+----3----+----4----+----
0000 SYSTEMD.UNIT=EMERGENCY.TARGET
Ready; T=0.01/0.01 09:12:33
```

To load systemd targets from the IPL command when you use DASD extended count key data (ECKD) or fixed-block architecture (FBA), use the following commands from the 3270 terminal:

```
===> ipl 100 PARM systemd.unit=rescue.target
00: zIPL v2.6.0-28.el8 interactive boot menu
00:
00:  0. default (Red Hat Enterprise Linux (4.18.0-193.el8.s390x) 8.2 (Ootpa))
00:
00:  1. Red Hat Enterprise Linux (4.18.0-193.el8.s390x) 8.2 (Ootpa)
00:  2. Red Hat Enterprise Linux (0-rescue-95742106573a4215ac86da4a36c81e40) 8.2
(Ootpa)
00: Note: VM users please use '#cp vi vmsg <input>'
00: Please choose (default will boot in 5 seconds):
...
You are in rescue mode. After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, "systemctl default" or "exit"
to boot into default mode.
Give root password for maintenance
(or press Control-D to continue):
```

If you want greater control when you IPL your Linux system, the initrd shell is a more flexible environment for advanced system administrators. To load the initrd shell, use the **rd.break** parameter instead of the **systemd.unit=** parameter; for example:

- When you use FCP Red Hat Enterprise Linux guests, use the following commands:

  ```
  ===> cp set loaddev portname 50050763 09141145 lun 40014022 00000000
  ===> cp set loaddev scpdata NEW 'rd.break'
  ===> ipl b747
  ```

- When you use DASD ECKD/FBA Red Hat Enterprise Linux guests, use the following command:

```
===> ipl 100 PARM rd.break
```

Important: If you need to recover the `root` password, the `rescue.target` and the `emergency.target` *cannot* help you with this task.

If you need to recover the `root` password for FCP Red Hat Enterprise Linux guests, complete the following steps:

1. Load the proper parameters from a 3270 terminal by using the following commands:

```
===> cp set loaddev portname 50050763 09141145 lun 40014022 00000000
===> cp set loaddev scpdata NEW 'rd.break'
```

2. IPL the virtual FCP device:

```
===> ipl b747
...
[ [0;32m  OK   [0m] Started Setup Virtual Console.
         Starting Dracut Emergency Shell...

Generating "/run/initramfs/rdsosreport.txt"


Entering emergency mode. Exit the shell to continue.
Type "journalctl" to view system logs.
You might want to save "/run/initramfs/rdsosreport.txt" to a USB stick or /boot
after mounting them and attach it to a bug report.

switch_root:/#
```

3. The system's `/sysroot` is mounted as read-only; remount it as read/write:

```
switch_root:/# mount -o remount,rw /sysroot
mount -o remount,rw /sysroot
```

4. Change the `root` password:

```
switch_root:/# chroot /sysroot passwd
chroot /sysroot passwd
Changing password for user root.
New password:
...
passwd: all authentication tokens updated successfully.
switch_root:/#
```

5. Instruct SELinux to relabel all files on restart because the `/etc/shadow` file was changed outside of its regular SELinux context:

```
switch_root:/# chroot /sysroot touch /.autorelabel
```

To exit and resume the IPL process, enter `exit`.

If you need to recover the `root` password for DASD ECKD/FBA Red Hat Enterprise Linux guests, complete the following steps to recover the `root` password:

1. IPL the virtual FCP device:

```
===> ipl 100 PARM rd.break
...
[ [0;32m  OK   [0m] Started Setup Virtual Console.
         Starting Dracut Emergency Shell...
```

```
Generating "/run/initramfs/rdsosreport.txt"

Entering emergency mode. Exit the shell to continue.
Type "journalctl" to view system logs.
You might want to save "/run/initramfs/rdsosreport.txt" to a USB stick or /boot
after mounting them and attach it to a bug report.

switch_root:/#
```

2. The system's `/sysroot` is mounted as read-only; remount it as read/write:

```
switch_root:/# mount -o remount,rw /sysroot
mount -o remount,rw /sysroot
```

3. Change the `root` password:

```
switch_root:/# chroot /sysroot passwd
chroot /sysroot passwd
Changing password for user root.
New password:
...
passwd: all authentication tokens updated successfully.
```

4. Instruct SELinux to relabel all files on reboot because the `/etc/shadow` file was changed outside of its regular SELinux context:

```
switch_root:/# chroot /sysroot touch /.autorelabel
```

5. To exit and resume the IPL process, enter `exit`.

11.1.2 Entering a rescue environment mode with Red Hat Enterprise Linux

If you encounter errors when you mount the root file system or experience other problems that prevent you from entering systemd targets, you can enter a *rescue environment*. This environment loads a Linux image in memory to enable critical repairs to the system.

To enter a rescue environment, start an interactive Linux installation. For example, complete the following steps to enter a rescue environment on the LINUX2 virtual machine:

1. Log on to the target Linux system by using the 3270 terminal console: for example, LINUX2.

   For the question "DO YOU WANT TO IPL LINUX FROM B747 AS STORAGE STORAGE WWPN 50050763 09141145 AT LOOKUP NUMBER 40014022 00000000? Y/N" that is asked by the REXX script that is loaded by the PROFILE EXEC, answer **n**:

```
LOGON LINUX2
00: NIC 0600 is created; devices 0600-0602 defined
00: NIC 0620 is created; devices 0620-0622 defined
00: NIC 0640 is created; devices 0640-0642 defined
00: z/VM Version 7 Release 1.0, Service Level 2001 (64-bit),
00: built on IBM Virtualization Technology
00: There is no logmsg data
00: FILES: 0003 RDR,   NO PRT,   NO PUN
00: LOGON AT 15:29:02 EDT WEDNESDAY 10/21/20
00: Command complete
00: Command complete
z/VM V7.1.0    2020-03-13 12:46
00: RPIMGR031E RESOURCE LNXADMIN SPECIFIED BY SPOOL COMMAND NOT FOUND
LINUX2   AT RDBKZVM3 VIA *        2020-10-21 15:29:02 EDT      WEDNESDAY
DMSACP723I T (592) R/O
```

```
DMSACR723I D (LNX:LNXADMIN.) R/O
DMSACR723I B (LNX:LNXADMIN.REDHAT82) R/O
DMSACR723I C (LNX:LNXADMIN.SWAPGEN) R/O
SWPSWA0012I DIAG swap disk defined at virtual address 0300 with
SWPSWA0012I 64988 4K pages of swap space.
SWPSWA0012I DIAG swap disk defined at virtual address 0301 with
SWPSWA0012I 129980 4K pages of swap space.
DO YOU WANT TO IPL LINUX FROM B747 AS
STORAGE WWPN 50050763 09141145  AT LOOKUP NUMBER 40014022 00000000? Y/N
n
```

2. Create a copy of the **LINUX1 PRM** file on the same local disk **LNX:LINUX2**. Rename it to RESCUE PRM:

   ```
   ===> copyfile linux1 prm a rescue prm a
   ```

3. Create a copy of the **REDHAT EXEC** file from **LNX:LNXADMIN.** to the local disk **LNX:LINUX1**. Rename it to RESCUE EXEC:

   ```
   ===> copyfile redhat exec a rescue exec a
   ```

4. Edit the RESCUE EXEC file and replace the GENERIC PRM with the RESCUE PRM parameter:

   ```
   /* */
   'CL RDR'
   'PURGE RDR ALL'
   'SPOOL PUNCH * RDR'
   'PUNCH KERNEL  IMG * (NOH'
   'PUNCH RESCUE PRM * (NOH'
   'PUNCH INITRD  IMG * (NOH'
   'CH RDR ALL KEEP NOHOLD'
   'I 00C'
   ```

5. Edit the GENERIC PRM file and replace any kickstart or Virtual Network Computing (VNC) lines with the **rescue** command-line option:

   ```
   ===> x rescue prm a
   ro ramdisk_size=47000 cio_ignore=all,!condev
   inst.repo=http://lnxadmin:lnx4rdbk@129.40.23.88/rhel82
   rd.znet=qeth,0.0.0640,0.0.0641,0.0.0642,layer2=1,portno=0,portname=Z
   ip=129.40.23.144::129.40.23.254:24:linux2.pbm.ihost.com:enc640:none
   nameserver=129.40.106.1 nameserver=129.40.106.2
   rd.zfcp=0.0.b747,0x5005076309141145,0x4001402200000000
   inst.cmdline
   rescue
   ```

6. Run the **RESCUE EXEC** to start the rescue environment:

   ```
   ===> rescue exec a
   00: 0000003 FILES CHANGED
   [    0.101778] Linux version 4.18.0-193.el8.s390x
   (mockbuild@s390-018.build.eng.bos.redhat.com) (gcc version 8.3.1 20191121 (Red
   Hat
    8.3.1-5) (GCC)) #1 SMP Fri Mar 27 14:43:09 UTC 2020
   [    0.101781] setup: Linux is running as a z/VM guest operating system in
   64-bit mode
   [    0.101841] setup: The maximum memory size is 8192MB
   [    0.101843] cma: Reserved 4 MiB at 0x00000001ffc00000
   [    0.101867] numa: NUMA mode: plain
   [    0.102019] cpu: 2 configured CPUs, 0 standby CPUs
   [    0.102155] Write protected kernel read-only data: 9456k
   ```

```
[    0.102204] Zone ranges:
[    0.102205]   DMA      [mem 0x0000000000000000-0x000000007fffffff]
[    0.102207]   Normal   [mem 0x0000000080000000-0x00000001ffffffff]
[    0.102208] Movable zone start for each node
[    0.102209] Early memory node ranges
[    0.102210]   node   0: [mem 0x0000000000000000-0x00000001ffffffff]
[    0.102212] Initmem setup node 0 [mem 0x0000000000000000-0x00000001ffffffff]
[    0.159615] percpu: Embedded 34 pages/cpu s99584 r8192 d31488 u139264
[    0.159644] Built 1 zonelists, mobility grouping on.  Total pages: 2064384
[    0.159644] Policy zone: Normal
[    0.159646] Kernel command line: ro ramdisk_size=47000
cio_ignore=all,!condev                                    inst.repo=http:/
/lnxadmin:lnx4rdbk@129.40.23.88/rhel82
rd.znet=qeth,0.0.0640,0.0.0641,0.0.0642,layer2=1,portno=0,portname=Z
      ip=129.40.23.144::129.40.23.254:24:linux2.pbm.ihost.com:enc640:none
nameserver=129.40.106.1 nameserver=129.4
0.106.2
rd.zfcp=0.0.b747,0x5005076309141145,0x4001402200000000
inst.cmdline
                                                                      rescue
...
Reloading system manager configuration
Starting default target
[   31.266141] anaconda[2036]: Starting installer, one moment...
[   31.266235] anaconda[2036]: 19:56:43 Please ssh install@linux2.pbm.ihost.com
(129.40.23.144) to begin the install.
```

The rescue process directs you to a Secure Shell (SSH) to connect to the IP address of your Linux server to begin the rescue environment.

7. Use an SSH client to connect to the IP address and log in as user `install`.
8. In the next instructions, the rescue environment prompts you to continue to mount the partitions that it finds on `/mnt/sysimage`, mount the found partitions as Read-Only, or skip this step:

```
Starting installer, one moment...
anaconda 29.19.2.17-1.el8 for Red Hat Enterprise Linux 8.2 started.
 * installation log files are stored in /tmp during the installation
 * shell is available in second TMUX pane (ctrl+b, then press 2)
 * if the graphical installation interface fails to start, try again with the
   inst.text bootoption to start text installation
 * when reporting a bug add logs from /tmp as separate text/plain attachments
================================================================================
=
================================================================================
=
Rescue

The rescue environment will now attempt to find your Linux installation and
mount it under the directory : /mnt/sysimage.  You can then make any changes
required to your system.  Choose '1' to proceed with this step.
You can choose to mount your file systems read-only instead of read-write by
choosing '2'.
If for some reason this process does not work choose '3' to skip directly to a
shell.

1) Continue
```

```
2) Read-only mount
3) Skip to shell
4) Quit (Reboot)

Please make a selection from the above:
```

Note: If you use PuTTY, under **Settings** → **Window** → **Translation**, change `Character Set Translation on Received Data` to `UTF-8` for a better user experience.

9. Choose **1** to continue. The following instructions appear:

```
================================================================================
================================================================================
Rescue Shell

Your system has been mounted under /mnt/sysimage.

If you would like to make the root of your system the root of the active
system,
run the command:

        chroot /mnt/sysimage

When finished, please exit from the shell and your system will reboot.
Please press ENTER to get a shell:

  <Tab>/<Alt-Tab> between elements   |  <Space> selects   |  <F12> next screen
[anaconda] 1:main* 2:shell  3:log  4:storage-log  5:program-log
```

10. Press **Enter**.

    Your system was mounted under `/mnt/sysimage`.

The rescue environment presents a command-line prompt:

```
sh-4.4#

When finished please exit from the shell and your system will reboot.
```

Note: If you want to use system tools to modify your system, use the `chroot` command line:

```
sh-4.4# chroot /mnt/sysimage/
bash-4.4# ls
bin   dev  home  lib64  mnt  proc  run   serial-getty@hvc0.service  sys  usr
boot  etc  lib   media  opt  root  sbin  srv                        tmp  var
```

To exit the rescue mode, enter `exit`. Return to the 3270 console to enter the following command; otherwise, the rescue environment is loaded again:

```
===> #CP IPL CMS
```

Note: If the rescue image cannot find your partition, you can attempt to mount it manually by using the **mount** command; for example, if the Linux system did not mount logical volumes (LVs) automatically.

To mount failed Logical Volume Manager (LVM) systems, enter the following commands:

```
bash-4.4# vgchange -ay
  1 logical volume(s) in volume group "vg_system" now active

bash-4.4# vgs
  VG        #PV #LV #SN Attr   VSize   VFree
  vg_system   1   1   0 wz--n- <59.00g    0

bash-4.4#  ls /dev/mapper/
control  live-base  live-rw  mpatha  mpatha1  mpatha2  vg_system-root

sh-4.4# mount /dev/mapper/vg_system-root /mnt/sysimage

sh-4.4# ls /mnt/sysimage/
bin   dev  home  lib64  mnt  proc  run   serial-getty@hvc0.service  sys  usr
boot  etc  lib   media  opt  root  sbin  srv                        tmp  var
```

Now, you can modify the files on the `vg_system-root` logical volume.

11.2 Setting up Memory Hotplugging

With Linux *Memory Hotplug*, you can increase or decrease the amount of memory in a Linux system without a reboot. Standby memory must be defined to the virtual machine in which Linux is running. You can issue the **CP DEFINE STORAGE** command to configure standby memory (storage).

To give the virtual machine another 1 GB of standby memory, you can change the directory or each virtual machine by using Conversational Monitor System (CMS) command-line mode. The following example shows how to change a specific Linux virtual machine to define standby memory by using the CMS command. The second example shows how to change the directory and how the Linux guest loads the changes.

To set up standby storage for Linux Memory Hotplug by using `LINUX1` as the virtual machine (for example), complete the following steps:

1. If you want this change to be temporary for this bootup of Linux only, or if you decide not to change your virtual machine's directory entry for another reason, type the following **DEFINE** statement from a 3270 terminal:

```
===> define storage 1GB standby 1GB
00: HCPZPM003E Invalid option - 1GB
Ready(00003); T=0.01/0.01 14:53:04
define storage 1G  standby 1G
00: STORAGE = 1G MAX = 2G INC = 2M STANDBY = 1G RESERVED = 0
00: Storage cleared - system reset.
===> ipl 100
```

2. If you want this change to be permanent, you must update the user directory entry for this virtual machine. For more information, see 4.12.3, "z/VM User Directory PROFILEs" in *The Virtualization Cookbook for IBM z Systems Volume 1: IBM z/VM 6.3,* SG24-8147.

3. After you change the directory, log on to `LINUX1`. The standby memory is reported:

```
LOGON LINUX1
00: z/VM Version 6 Release 3.0, Service Level 1501 (64-bit),
00: built on IBM Virtualization Technology
00: There is no logmsg data
00: FILES: 0003 RDR,   NO PRT,   NO PUN
00: LOGON AT 15:27:29 EDT MONDAY 04/27/15
00: STORAGE = 1G MAX = 2G INC = 2M STANDBY = 1G RESERVED = 0
00: Storage cleared - system reset.
00: Command complete
00: NIC 0600 is created; devices 0600-0602 defined
00: NIC 0600 is connected to VSWITCH SYSTEM VSW1
z/VM V6.3.0    2015-04-09 09:04
LINUX1   AT ITSOZVM1 VIA RSCS     2015-04-27 15:27:30 EDT      MONDAY
DMSACR723I D (LNX:LNXADMIN.) R/O
DIAG swap disk defined at virtual address 0300 (64988 4K pages of swap space)
DIAG swap disk defined at virtual address 0301 (129980 4K pages of swap space)
DO YOU WANT TO IPL LINUX FROM MINIDISK 100? Y/N
Y
```

4. Start an SSH session as `root` and view the memory in the `/sys/` file system. Change the directory to `/sys/devices/system/memory/` and list the files:

```
# cd /sys/devices/system/memory
# ls
block_size_bytes  memory1  memory3  memory5  memory7  uevent
```

```
memory0           memory2  memory4  memory6  power
```

5. Enter the `block_size_bytes` file by using the **`cat`** command:

```
# cat block_size_bytes
10000000
```

This number is the number of bytes in hexadecimal. 10000000 in hex is 256 MB in decimal. Therefore, the block size is 256 MB and eight blocks (`memory0 - memory7`) exist, which are represented as directories. Each memory block has a state, which is represented as a file.

6. Show the state of each memory block with the following command:

```
# cat memory*/state
online
online
online
online
offline
offline
offline
offline
```

This output shows that the first 1 GB is online and the next four blocks are offline.

7. You also can show information about memory by using the **`free -m`** command:

```
# free -m
              total        used        free      shared  buff/cache   available
Mem:          991          94           766      0          130          858
Swap:         1785        0          1785
```

This output shows 766 MB of free memory that is available (part of the memory is used internally by Linux).

8. You can turn on memory by sending the string *online* to the state file. Turn on another 512 MB of memory with the following commands:

```
# echo online > memory4/state
# echo online > memory5/state
```

9. Show that the memory is now online:

```
# cat memory*/state
online
online
online
online
online
online
offline
offline
```

10. Again, confirm the available memory with the **`free -m`** command:

```
# free -m
               total        used        free      shared  buff/cache   available
Mem:            1503          96        1276           0         130        1367
Swap:           1785           0        1785
```

This output shows that 1,276 MB of free memory is now available.

11. You can also return the memory by echoing *offline* to the state file:

```
# echo offline  > memory4/state
# echo offline  > memory5/state
```

12. Verify that the memory was returned:

```
# cat memory*/state
online
offline
offline
offline
# free -m
              total        used        free      shared  buff/cache   available
Mem:            991          94         768           0         129         859
Swap:          1785           0        1785
```

We demonstrated how to configure virtual machines with standby memory and how to hot-plug the memory from Linux. This function can increase your system's performance and availability.

11.3 Using the cpuplugd service

By using the *cpuplugd* service, Linux can enable or disable CPUs and memory, based on a set of rules. It can improve performance by setting the correct number of processors and the correct amount of memory for Linux systems, depending on their current loads. It can also prevent the Linux scheduler from queue balancing in partial load situations.

11.3.1 Determining the virtual CPUs that are used

To start working with **cpuplugd**, complete the following steps:

1. Start an SSH session to the Linux system and determine the number of CPUs that Linux has online. Use the command **lscpu**:

```
# lscpu
Architecture:        s390x
CPU op-mode(s):      32-bit, 64-bit
Byte Order:          Big Endian
CPU(s):              4
On-line CPU(s) list: 0-3
Thread(s) per core:  1
Core(s) per socket:  1
Socket(s) per book:  1
Book(s) per drawer:  1
Drawer(s):           4
NUMA node(s):        1
Vendor ID:           IBM/S390
Machine type:        8561
CPU dynamic MHz:     5200
CPU static MHz:      5200
BogoMIPS:            24038.00
Hypervisor:          z/VM 7.1.0
Hypervisor vendor:   IBM
Virtualization type: full
Dispatching mode:    horizontal
```

```
L1d cache:           128K
L1i cache:           128K
L2d cache:           4096K
L2i cache:           4096K
L3 cache:            262144K
L4 cache:            983040K
NUMA node0 CPU(s):   0-3
Flags:               esan3 zarch stfle msa ldisp eimm dfp edat etf3eh highgprs
te vx vxd vxe gs vxe2 vxp sort dflt sie
```

2. Observe the status of the **cpuplugd** service:

```
# systemctl status cpuplugd
cpuplugd.service - CPU hotplug daemon for Linux on System z
   Loaded: loaded (/usr/lib/systemd/system/cpuplugd.service; disabled; vendor
preset: disabled)
   Active: inactive (dead)
     Docs: man:cpuplugd(8)
           man:cpuplugd.conf(5)
```

3. Start the cpuplugd daemon:

```
# systemctl start cpuplugd
# systemctl status cpuplugd
? cpuplugd.service - CPU hotplug daemon for Linux on System z
   Loaded: loaded (/usr/lib/systemd/system/cpuplugd.service; disabled; vendor
preset: disabled)
   Active: active (running) since Thu 2020-10-01 08:51:43 EDT; 34s ago
     Docs: man:cpuplugd(8)
           man:cpuplugd.conf(5)
  Process: 629298 ExecStart=/usr/sbin/cpuplugd -c /etc/cpuplugd.conf
(code=exited, status=0/SUCCESS)
 Main PID: 629299 (cpuplugd)
    Tasks: 1 (limit: 101853)
   Memory: 580.0K
   CGroup: /system.slice/cpuplugd.service
           ··629299 /usr/sbin/cpuplugd -c /etc/cpuplugd.conf
```

4. Wait a few minutes and run the **lscpu** script again:

```
# lscpu
Architecture:         s390x
CPU op-mode(s):       32-bit, 64-bit
Byte Order:           Big Endian
CPU(s):               4
On-line CPU(s) list:  0
Off-line CPU(s) list: 1-3
Thread(s) per core:   1
Core(s) per socket:   1
Socket(s) per book:   1
Book(s) per drawer:   1
Drawer(s):            1
NUMA node(s):         1
Vendor ID:            IBM/S390
Machine type:         8561
CPU dynamic MHz:      5200
CPU static MHz:       5200
BogoMIPS:             24038.00
Hypervisor:           z/VM 7.1.0
```

```
Hypervisor vendor:    IBM
Virtualization type:  full
Dispatching mode:     horizontal
L1d cache:            128K
L1i cache:            128K
L2d cache:            4096K
L2i cache:            4096K
L3 cache:             262144K
L4 cache:             983040K
NUMA node0 CPU(s):    0
Flags:                esan3 zarch stfle msa ldisp eimm dfp edat etf3eh highgprs
te vx vxd vxe gs vxe2 vxp sort dflt sie
```

The output shows that now only one of the four virtual CPUs are active. The **cpuplugd** service turned off the other three virtual CPUs.

5. The **cpuplugd** configuration file is `/etc/cpuplugd.conf`. Specific middleware products recommend a minimum of two virtual processors. If most of your Linux servers run a workload that recommends two processors, change the default for `CPU_MIN` to `2`. The exception is when only a single physical processor is available. View the non-comments and lines that are not blank in the configuration file by using the following command:

```
# cd /etc
# egrep -v '^$|^#' cpuplugd.conf
CPU_MIN="1"
CPU_MAX="0"
UPDATE="1"
CMM_MIN="0"
CMM_MAX="131072"        # 512 MB
pgscan_d="vmstat.pgscan_direct_dma[0] + vmstat.pgscan_direct_normal[0] +
vmstat.pgscan_direct_movable[0]"
pgscan_d1="vmstat.pgscan_direct_dma[1] + vmstat.pgscan_direct_normal[1] +
vmstat.pgscan_direct_movable[1]"
pgscanrate="(pgscan_d - pgscan_d1) / (cpustat.total_ticks[0] -
cpustat.total_ticks[1])"
avail_cache="meminfo.Cached - meminfo.Shmem"
user_0="(cpustat.user[0] - cpustat.user[1])"
nice_0="(cpustat.nice[0] - cpustat.nice[1])"
system_0="(cpustat.system[0] - cpustat.system[1])"
user_2="(cpustat.user[2] - cpustat.user[3])"
nice_2="(cpustat.nice[2] - cpustat.nice[3])"
system_2="(cpustat.system[2] - cpustat.system[3])"
CP_Active0="(user_0 + nice_0 + system_0) / (cpustat.total_ticks[0] -
cpustat.total_ticks[1])"
CP_Active2="(user_2 + nice_2 + system_2) / (cpustat.total_ticks[2] -
cpustat.total_ticks[3])"
CP_ActiveAVG="(CP_Active0+CP_Active2) / 2"
idle_0="(cpustat.idle[0] - cpustat.idle[1])"
iowait_0="(cpustat.iowait[0] - cpustat.iowait[1])"
idle_2="(cpustat.idle[2] - cpustat.idle[3])"
iowait_2="(cpustat.iowait[2] - cpustat.iowait[3])"
CP_idle0="(idle_0 + iowait_0) / (cpustat.total_ticks[0] -
cpustat.total_ticks[1])"
CP_idle2="(idle_2 + iowait_2) / (cpustat.total_ticks[2] -
cpustat.total_ticks[3])"
CP_idleAVG="(CP_idle0 + CP_idle2) / 2"
CMM_INC="meminfo.MemFree / 40"
```

```
CMM_DEC="meminfo.MemTotal / 40"
HOTPLUG="((1 - CP_ActiveAVG) * onumcpus) < 0.08"
HOTUNPLUG="(CP_idleAVG * onumcpus) > 1.15"
MEMPLUG="0"
MEMUNPLUG="0"
```

The default rules for the plugging and unplugging of CPUs in the configuration file are shown:

```
HOTPLUG="((1 - CP_ActiveAVG) * onumcpus) < 0.08"
HOTUNPLUG="(CP_idleAVG * onumcpus) > 1.15"
```

The variables in the statements are defined:

CP_ActiveAVG	The current average CPU load
onumcpus	The number of online CPUs
runable_proc	The current number of processes that can be run
CP_idleAVG	The current idle percentage

These CPU hot plugging and unplugging values are used as described in 11.3.2, “Generating a workload to demonstrate cpuplugd”. In the default setup, **cpuplugd** changes the virtual processor configuration only. The auto-adaptive adjustment of the memory by using the `cmm` feature (module) is deactivated by default and is unavailable when you run in a native logical partition (LPAR) environment.

11.3.2 Generating a workload to demonstrate cpuplugd

You can now generate a workload to show how **cpuplugd** turns on CPUs.

Important: Running the following command generates significant CPU use. Verify that no mission-critical workload is running on this z/VM LPAR because this test might affect it. Also, ensure that you end the processes after you see **cpuplugd** work.

Complete the following steps:

1. Put 10 looping jobs in the background by using the following **for** loop:

```
# for i in `seq 1 10`
> do
>   bash -c "cat /dev/zero > /dev/null" &
> done
[1] 630071
[2] 630072
[3] 630073
[4] 630074
[5] 630075
[6] 630076
[7] 630077
[8] 630078
[9] 630079
[10] 630080
```

2. Verify that the jobs are running (the **top** command also can be used):

```
# pstree -G | grep cat
        ·      ··sshd···sshd···bash···10*[bash···cat]
```

3. Run the **lscpu** command. The following example shows that **cpuplugd** started the other spare processor after a few minutes:

```
# lscpu
Architecture:        s390x
CPU op-mode(s):      32-bit, 64-bit
Byte Order:          Big Endian
CPU(s):              4
On-line CPU(s) list: 0-3
Thread(s) per core:  1
Core(s) per socket:  1
Socket(s) per book:  1
Book(s) per drawer:  1
Drawer(s):           4
NUMA node(s):        1
Vendor ID:           IBM/S390
Machine type:        8561
CPU dynamic MHz:     5200
CPU static MHz:      5200
BogoMIPS:            24038.00
Hypervisor:          z/VM 7.1.0
Hypervisor vendor:   IBM
Virtualization type: full
Dispatching mode:    horizontal
L1d cache:           128K
L1i cache:           128K
L2d cache:           4096K
L2i cache:           4096K
L3 cache:            262144K
L4 cache:            983040K
NUMA node0 CPU(s):   0-3
Flags:               esan3 zarch stfle msa ldisp eimm dfp edat etf3eh highgprs
te vx vxd vxe gs vxe2 vxp sort dflt sie
```

After a few minutes, all of the CPUs are activated.

4. Stop the workload that was created before you use the **killall** command:

```
# killall cat
bash: line 1: 630084 Terminated              cat /dev/zero > /dev/null
bash: line 1: 630087 Terminated              cat /dev/zero > /dev/null
bash: line 1: 630089 Terminated              cat /dev/zero > /dev/null
bash: line 1: 630082 Terminated              cat /dev/zero > /dev/null
bash: line 1: 630085 Terminated              cat /dev/zero > /dev/null
bash: line 1: 630090 Terminated              cat /dev/zero > /dev/null
bash: line 1: 630088 Terminated              cat /dev/zero > /dev/null
bash: line 1: 630083 Terminated              cat /dev/zero > /dev/null
bash: line 1: 630086 Terminated              cat /dev/zero > /dev/null
bash: line 1: 630081 Terminated              cat /dev/zero > /dev/null

[1]   Exit 143               bash -c "cat /dev/zero > /dev/null"
[2]   Exit 143               bash -c "cat /dev/zero > /dev/null"
[3]   Exit 143               bash -c "cat /dev/zero > /dev/null"
[4]   Exit 143               bash -c "cat /dev/zero > /dev/null"
[5]   Exit 143               bash -c "cat /dev/zero > /dev/null"
[6]   Exit 143               bash -c "cat /dev/zero > /dev/null"
[7]   Exit 143               bash -c "cat /dev/zero > /dev/null"
[8]   Exit 143               bash -c "cat /dev/zero > /dev/null"
```

```
[9]-  Exit 143                bash -c "cat /dev/zero > /dev/null"
[10]+  Exit 143                bash -c "cat /dev/zero > /dev/null"
```

5. Run the **lscpu** command. The following example shows that **cpuplugd** stopped the other processor after a few minutes:

```
# lscpu
Architecture:          s390x
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Big Endian
CPU(s):                4
On-line CPU(s) list:   0
Off-line CPU(s) list:  1-3
Thread(s) per core:    1
Core(s) per socket:    1
Socket(s) per book:    1
Book(s) per drawer:    1
Drawer(s):             1
NUMA node(s):          1
Vendor ID:             IBM/S390
Machine type:          8561
CPU dynamic MHz:       5200
CPU static MHz:        5200
BogoMIPS:              24038.00
Hypervisor:            z/VM 7.1.0
Hypervisor vendor:     IBM
Virtualization type:   full
Dispatching mode:      horizontal
L1d cache:             128K
L1i cache:             128K
L2d cache:             4096K
L2i cache:             4096K
L3 cache:              262144K
L4 cache:              983040K
NUMA node0 CPU(s):     0
Flags:                 esan3 zarch stfle msa ldisp eimm dfp edat etf3eh highgprs
te vx vxd vxe gs vxe2 vxp sort dflt sie
```

11.3.3 Setting memory sizes by using cpuplugd

Memory sizes also can be set by the **cpuplugd** service. However, unlike CPUs, no good generic default value is available. The following example is extracted from a sample configuration that is available in *Linux on z Systems Device Drivers, Features and Commands*:

```
CMM_MIN="0"
CMM_MAX="131072"# 512 MB
pgscan_d="vmstat.pgscan_direct_dma[0] + vmstat.pgscan_direct_normal[0] +
vmstat.pgscan_direct_movable[0]"
pgscan_d1="vmstat.pgscan_direct_dma[1] + vmstat.pgscan_direct_normal[1] +
vmstat.pgscan_direct_movable[1]"
# page scan rate in pages / timer tick
pgscanrate="(pgscan_d - pgscan_d1) / (cpustat.total_ticks[0] -
cpustat.total_ticks[1])"
# cache usage in kilobytes
```

```
avail_cache="meminfo.Cached - meminfo.Shmem"
# cmm_inc: 10% of free memory, in 4K pages
CMM_INC="meminfo.MemFree / 40"
# cmm_dec: 10% of total memory, in 4K pages
CMM_DEC="meminfo.MemTotal / 40"
...
MEMPLUG="pgscanrate > 20"
MEMUNPLUG="(meminfo.MemFree + avail_cache) > (meminfo.MemTotal / 10)"
```

However, this example is only a starting point to explain the syntactical structure of a rule. Do not use this configuration in production. You must test any setting that you want to implement against a representative workload that your Linux systems to be running (the details are beyond the scope of this section).

For more information about **cpuplugd** on Red Hat Enterprise Linux 8.2, see IBM Documentation.

11.4 X Window System

For many years, UNIX operating systems used X Window System, which is also called “X”. This system was designed to provide a client/server, hardware-independent, and network-enabled graphical environment. Linux systems now use X.Org, which is an open source implementation of X Window System.

The X communication protocol by its nature is not secure at all. Therefore, X is often used with SSH protocol, which tunnels X11 traffic by using encrypted (and therefore secure) communications.

X11 displays graphics on a raster display. If the user wants to move, resize, and otherwise manage windows, a *window manager* is needed. Many window managers are available; several are lightweight and others are more robust. The use of a window manager is a good idea because it provides functions that are expected from a graphical user interface (GUI).

When Linux is installed on your workstation, a window manager is likely insufficient. You need a full desktop environment with menus, icons, taskbars, and so on, such as Gnome and KDE. Installing GNOME or KDE on IBM Z is discouraged because they are resource intensive.

11.4.1 Wayland

Starting with Red Hat Enterprise Linux 8, Red Hat used Wayland as the default display manager for GNOME, which replaced Xorg that was used by X11. The Wayland protocol uses the GNOME shell as the Wayland compositor and display server, unlike X11 where these components are separate. Wayland is simpler than X11 and is considered secure, which is one of the major reasons why Red Hat is moving towards it.

Displaying graphics by using Wayland protocol is different than with X11. Visible differences are found for users who are using the GNOME desktop interface to access their Red Hat Enterprise Linux systems.

Because of specific limitations with Wayland, and as not many vendors adopted this Wayland yet, X11 continues to be used. Red Hat provides switching back to X11 for its GNOME interface and another XWayland server, which acts as a translator for X11 applications to communicate with the Wayland compositor.

XWayland acts as a proxy between established X11 clients and the Wayland compositor, functioning as X11 server and Wayland client.

11.4.2 VNC server

The X server is run on the workstation where the mouse, keyboard, and monitor are used. The VNC server provides a virtual workstation with all of these peripherals (virtual). The VNC server starts an embedded X server. Then, any X-based application can send its output to this X server, whether the applications are local or remote to the X server.

VNC is session-oriented, which is an advantage. If communication to the VNC server is lost, a new connection is reestablished to the session as it was. Also, applications in a disconnected VNC session still continue to run.

For more information about setting up a VNC server on Red Hat Enterprise Linux, see 1.2.7, “Configuring the Virtual Network Computing server” on page 24.

11.4.3 Using embedded SSH to forward X with Red Hat Enterprise Linux

If VNC is not acceptable, you can use a standard X server on a workstation. Because Linux users usually know the X Window System, an X server that runs on Windows is described in this section.

To forward X on SSH connections, use the following command to install the packages needed by the target system:

```
# yum install -y xorg-x11-xauth xorg-x11-fonts-* xorg-x11-utils
```

To connect to the target system, use the following command from your Linux workstation:

```
# ssh -l root -X <target_system>
```

Using PuTTY

To use PuTTY for X11 forwarding, select **Enable X11 forwarding**, as shown in Figure 11-1.

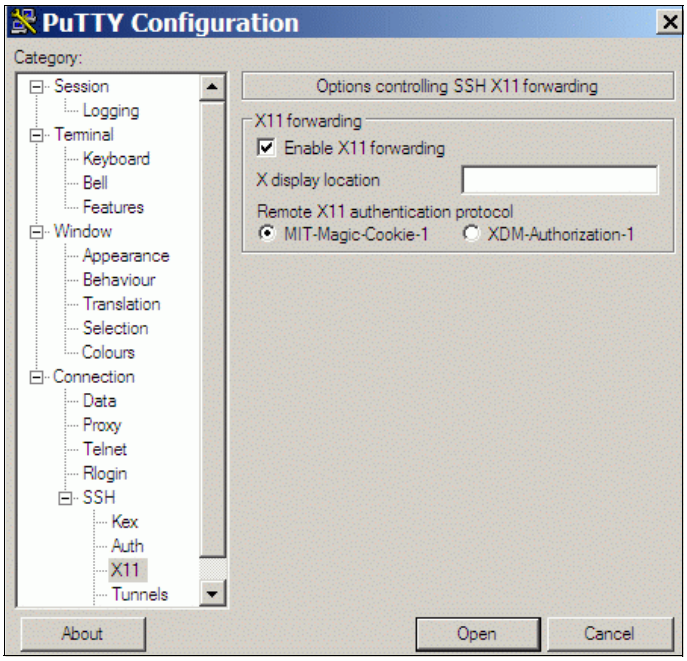

Figure 11-1 Allow X11 forwarding in PuTTY

When you are connected to a remote Linux system with X11 forwarding enabled, the `DISPLAY` environment variable contains the special value of `localhost:10.0`, which tells PuTTY to forward X11 protocol over SSH to the SSH client address. PuTTY requires an X Window Server that is running on your Microsoft Windows workstation.

Many commercial and free X Window servers are available for Microsoft Windows, which provides a free X server that is based on Cygwin.

You can achieve the same results by using several methods. It is up to you to choose a solution that suits your purpose best.

11.5 Setting up the IUCV Linux Terminal Server

Implementation of a Linux Terminal Server (LTS) that is based on z/VM Inter User Communications Vehicle (IUCV) allows access to the Linux console without a functioning TCP/IP stack on Linux.

Many IBM customers who run Linux under z/VM consider the implementation of this IUCV LTS to be a fundamental requirement for a Linux Virtual Server to be eligible for classification as a production system in their environment. Also, IUCV LTS is based on a character mode interface, which enables the use of traditional Linux full-screen tools, such as **`vi`**.

For more information about official documentation of this IUCV LTS setup, see IBM Documentation.

Although the documentation offers many options to set up the IUCV LTS, this section describes the implementation of the **`iucvtty`** command.

Implementation of the LTS includes topics that involve changes on z/VM and Linux.

11.5.1 Red Hat Enterprise Linux configuration for IUCV Linux Terminal Server

The **IUCV ALLOW** line allows virtual machines to connect to other virtual machines, such as the LTS, by using IUCV. The `IUCV ALLOW` line was included in the LNXPDFLT PROFILE entry, which is described in 6.13.1, “z/VM User directory profiles” of *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147.

The LTS configuration is described in 1.2.5, “Configuring IUCV Linux Terminal Server” on page 20.

To enable the target Linux system to receive IUVC connections, enable the service as described in the post installation section of 1.2.6, “Configuring Kickstart” on page 21.

Run the following commands to enable `hvc0` as a serial console:

```
# ln -s /etc/systemd/system/serial-getty\@hvc0.service
/lib/systemd/system/serial-getty\@.service
```

11.6 Issuing z/VM CP commands from Linux

The **vmcp** command allows z/VM control program (CP) commands to be issued from Linux; for example:

```
# vmcp query v dasd
DASD 0100 3390 VV1569 R/W      10016 CYL ON DASD  1569 SUBCHANNEL = 0000
DASD 0120 3390 VV1560 R/O        140 CYL ON DASD  1560 SUBCHANNEL = 000D
DASD 0190 3390 VV1560 R/O        214 CYL ON DASD  1560 SUBCHANNEL = 0006
DASD 0191 3390 VV1560 R/O        500 CYL ON DASD  1560 SUBCHANNEL = 0009
DASD 019D 3390 VV1560 R/O        292 CYL ON DASD  1560 SUBCHANNEL = 0007
DASD 019E 3390 VV1560 R/O        500 CYL ON DASD  1560 SUBCHANNEL = 0008
DASD 0200 3390 VV156B R/W      10016 CYL ON DASD  156B SUBCHANNEL = 0001
DASD 0300 9336 (VDSK) R/W     524288 BLK ON DASD  VDSK SUBCHANNEL = 000E
DASD 0301 9336 (VDSK) R/W    1048576 BLK ON DASD  VDSK SUBCHANNEL = 000F

# vmcp query names
LINUX3   - SSI , LINUX4   - SSI , DIRMSAT2 - SSI
DATAMOVE - DSC , MAINT630 -LO006, LNXSERV1 - DSC , DIRMAINT - DSC
FTPSERVE - DSC , SSLSERVE - DSC , TCPIP    - DSC , ZHCP     - DSC
XCAT     - DSC , VSMEVSRV - DSC , VSMREQIU - DSC , VSMREQI6 - DSC
VSMREQIN - DSC , DTCSMAPI - DSC , PERSMAPI - DSC , VSMWORK3 - DSC
VSMWORK2 - DSC , VSMWORK1 - DSC , VSMGUARD - DSC , LNXADMIN -LO007
DTCVSW2  - DSC , DTCVSW1  - DSC , VMSERVP  - DSC , VMSERVR  - DSC
VMSERVU  - DSC , VMSERVS  - DSC , OPERSYMP - DSC , DISKACNT - DSC
EREP     - DSC , OPERATOR - DSC , MAINT    -LO005, LINUX2   -LO009
VSM     - TCPIP

# vmcp query v storage
STORAGE = 16G

# vmcp query vswitch details
VSWITCH SYSTEM VSW1     Type: QDIO    Connected: 3    Maxconn: INFINITE
  PERSISTENT  RESTRICTED    ETHERNET                  Accounting: OFF
```

```
  USERBASED LOCAL
  VLAN Unaware
  MAC address: 02-00-0A-00-00-01    MAC Protection: Unspecified
...
```

For more information about CP commands, see *z/VM V7.2 CP Commands and Utilities Reference*, SC24-6268.

11.7 Accessing z/VM CMS disks from Linux

You can access the data on the z/VM CMS disk by using the CMS file system tools. Alternatively, the CMS disk can be mounted by using `cmsfs-fuse`.

11.7.1 Using the CMS file system tools

To use the CMS file system tools, the following package must be installed for Red Hat Enterprise Linux:

```
# yum install s390utils-cmsfs
```

Complete the following steps:

1. Set the device with the CMS file system online:

```
# cio_ignore -r 0.0.0190
# chccwdev -e 0.0.0190
Setting device 0.0.0190 online
Done
# lsdasd 0.0.0190
Bus-ID    Status    Name      Device  Type         BlkSz  Size      Blocks
==============================================================================
0.0.0190  active(ro)  dasdd     94:12   ECKD         4096   150MB     38520
```

2. Print the table of contents of that CMS disk:

```
# cmsfslst -d /dev/dasdd | head
FILENAME FILETYPE FM FORMAT LRECL       RECS     BLOCKS     DATE     TIME
         DIRECTOR PO F         64        713          1  4/19/2020  8:46:00
         ALLOCMAP PO F       4096          2          2  4/19/2020  9:46:00
$$LDR$$  XEDIT    M2 V         63        243          4  6/22/2018 12:35:43
$@VML$NK SXEDIT   L2 V         63        373          4  6/22/2018 12:23:31
$@VML$NK XEDIT    L2 F       1024         34          9  6/22/2018 14:03:41
$DASD$   CONSTS   L2 V         63        111          1  6/22/2018 12:21:54
$DUP     EXEC     L2 V         55         74          1  6/22/2018 12:22:05
$MOVE    EXEC     L2 V         80        197          1  6/22/2018 12:22:05
$VMFCPM$ EXEC     M2 V        120        758          7  6/22/2018 12:35:39
```

3. Show the content of a file:

```
# cmsfscat -d /dev/dasdd -a __CPL.H | head
                  ??=ifndef __sys_cpl
                  ??=ifdef __COMPILER_VER__
                    ??=pragma filetag("IBM-1047")
                  ??=endif
                  #define __sys_cpl 1
                  #pragma nomargins nosequence
                  #pragma checkout(suspend)
```

```
/*****************************************************************
* <sys/__cpl.h> header file                                      *
*                                                                *
```

4. Set the device with the CMS file system offline:

```
# chccwdev -d 0.0.0190
Setting device 0.0.0190 offline
Done
```

11.7.2 Mounting a CMS disk by using cmsfs-fuse

To mount the CMS file system tools, the following packages must be installed for Red Hat Enterprise Linux:

```
# yum install fuse s390utils-cmsfs-fuse
```

Complete the following steps:

1. Set the device with the CMS file system online:

```
# cio_ignore -r 0.0.0190
# chccwdev -e 0.0.0190
Setting device 0.0.0190 online
Done
# lsdasd 0.0.0190
Bus-ID    Status    Name      Device  Type         BlkSz  Size      Blocks
==============================================================================
0.0.0190  active(ro)  dasdd     94:12   ECKD         4096   150MB     38520
```

2. Mount the CMS disk to `/mnt`:

```
# cmsfs-fuse -a -o ro /dev/disk/by-path/ccw-0.0.0190 /mnt
```

3. Access data:

```
# ls /mnt/EDIT.EXEC
/mnt/EDIT.EXEC

# tail -n5 /mnt/EDIT.EXEC
&LOOP 3 &K
&IF &&I = LRECL  &&I = WIDTH
&IF &&I = NODISP &&I = NOSCREEN
&I = &I + 1
&GOTO -GO
```

4. Unmount the CMS disk:

```
# fusermount -u /mnt
```

5. Set the device with the CMS file system offline:

```
# chccwdev -d 0.0.0190
Setting device 0.0.0190 offline
Done
```

11.8 Network File System mounting the LNXADMIN SFS directory from Linux

Although clients seldom perform this task, the authors of this book chose to include this topic for reference. If you use this function, we recommend that you unmount as soon as it is no longer required so that Shared File System (SFS) performance is not affected. If too many unnecessary NFS mounts are open to the file pool, SFS performance might be affected.

Part 3

Appendixes

This section consists of the following appendixes:

A

Reference sheets, cheat sheets, and blank worksheets

This appendix refers to additional materials that are included for your reference, which can be printed or downloaded from the internet as described.

This appendix includes the following topics:

Important z/VM files

z/VM differs from Linux in the location and number of configuration files. In Linux, many configuration files are in Linux and most of them are in or under the `/etc/` directory.

Few configuration files are in z/VM; however, they are on many different minidisks. Table A-1 provides a summary and the location of important z/VM configuration files.

Table A-1 Important z/VM configuration files

File	Location	Description
`SYSTEM CONFIG`	`PMAINT CF0`	This file is the operating system's main configuration file. It defines the system name, control program (CP) volumes, user volumes, and other settings.
`USER DIRECT`	`MAINT 2CC`	This file is the initial z/VM user directory. All virtual machines that are known to the system are defined here. If a directory maintenance product is in use, this file is no longer authoritative.
`PROFILE TCPIP`	`TCPMAINT 198`	This file defines the resources for the primary z/VM TCP/IP stack, including the TCP/IP address, Open Systems Adapter (OSA) resources, subnet mask, and gateway. It is initially created by the IPWIZARD tool as `PROFILE TCPIP`.
`SYSTEM DTCPARMS`	`TCPMAINT 198`	This file is created to define the TCP/IP stacks on the system. It is initially created by the IPWIZARD tool.
`TCPIP DATA`	`TCPMAINT 592`	This file defines the Domain Name System (DNS) server, the domain name, and other settings. It is initially created by the IPWIZARD tool.
`PROFILE EXEC`	`AUTOLOG1 191`	This file is a REXX EXEC that is run when the system starts. It is analogous to the `/etc/inittab` file in Linux.

Cheat sheets

This section contains quick references, or "cheat sheets", for the XEDIT and `vi` editors.

XEDIT cheat sheet

`XEDIT` features line commands that are entered on the command line (====>) and prefix commands, which are typed over the line numbers on the left side of the window.

Line commands

Do not include the < > in your commands:

`a` Add a line.

`a<n>` Add <n> lines.

`c/<old>/<new>/<n> <m>`

Search for string <old> and replace it with <new> for <n> lines below the current line and <m> times on each line. An asterisk (*) can be used for <n> and <m>.

`/<string>` Search for 'string' from the current line.

-/<string>	Search backwards for 'string'.
all /<string>/	Show all occurrences of 'string' and hide other lines.
bottom	Move to the bottom of the file.
top	Move to the top of the file.
down <n>	Move down 'n' lines.
up <n>	Move up 'n' lines.
file	Save the current file and exit XEDIT.
ffile	Save the current file and exit but do not warn of overwrite.
save	Save the current file but do not exit.
quit	Exit XEDIT if no changes were made.
qquit	Exit XEDIT, even if changes were not saved.
left <n>	Shift 'n' characters to the left.
right <n>	Shift 'n' characters to the right.
get <file>	Copy file and insert past the current line.
input	Enable INPUT mode to insert multiple lines of text, beginning at the current line.
:<n>	Move to line 'n'.
?	Display last command.
=	Execute last command.
x <file>	Edit 'file' and put it into the XEDIT “ring”.
x	Move to the next file in the ring.

Prefix commands

The following commands are prefix commands:

a	Add one line.
a<n>	Add 'n' lines.
c	Copies one line.
cc	Copies a block of lines.
d	Deletes one line.
dd	Deletes a block of lines.
f	Line after which a copy (c) or a move (m) is to be inserted.
p	Line before which a copy (c) or a move (m) is to be inserted.
i	Insert a line.
i<n>	Insert 'n' lines.
m	Move one line.
mm	Move a block of lines.
"	Replicate a line.
"<n>	Replicate a line 'n' times.
""	Replicate a block of lines.

A vi cheat sheet

The following small subset of **vi** commands shows the most commonly used commands. The vi editor has three modes:

- Input mode: The **Insert** key, **i**, **o** (add a line below), **0** (add a line above), and other commands enter you into this mode where you can type text into the file. When you are in this mode, you see the text --INSERT-- in the last line.
- Command mode: Pressing Esc gets you out of input mode and into command mode. You can issue the following commands:

i	Brings you back to input mode.
dd	Deletes a line and puts it in the buffer.
<n>dd	Delete 'n' lines.
x	Delete a character.
dw	Delete a word.
p	Add the buffer past the current location.
P	Add the buffer before the current location.
o	Add a line and go into insert mode.
/string	Search for string.
n	Repeat the last command, which can be powerful.
jkl;	Cursor movement.
A	Add text at the end of the line.
<nn>G	Go to line 'nn'.
G	Go to the last line in the file.
yy	Yank a line (copy into buffer).
<n>yy	Yank 'n' lines.

- Command-line mode: Pressing the colon (**:**) key brings you to this mode at the bottom of the window. You can issue the following commands:

:wq	Ssave (write and quit).
:q!	Quit and discard changes.
:<nn>	Go to line number 'nn'.
:r <file>	Read 'file' into the current file.
:1,$s/old/new/g	Globally replace 'old' with 'new'.
:help	Give help.

DirMaint cheat sheet

The following DirMaint commands are available:

Add	Add a user or profile directory entry.
AMDisk	Add a minidisk.
DEDicate	Add or delete an existing dedicate statements.
DMDisk	Remove a minidisk.
FILE	Add or replace a DirMaint control file.
RLDCode	Reload DirMaint resident operating procedures.
RLDExtn	Reload DirMaint CONFIG* DATADVH file.
REView	Review a user or profile directory entry.
MDisk	Change the access mode and passwords for minidisks.
STorage	Change the log on storage size.
SEND	Request a copy of a DirMaint control file.
SETOptn	Add, change, or delete CP options.
CLAss	Change the CP class for a directory entry.
SPEcial	Add or delete an existing special statement.

DirMaint example commands

The following commands are examples of DirMaint commands:

- Add a 50 cylinder minidisk 200 to user ID spiedie:

```
DIRMAINT FORUSER SPIEDIE AMDISK 200 3390 AUTOG 50 {VOLGROUP}
```

- Add a link statement to the TCPMAINT 592 minidisk into the directory entry for user vmfrau:

```
DIRMAINT FORUSER VMFRAU LINK TCPMAINT 0592 0592 RR
```

Blank planning worksheet

This section contains a blank copy of the planning worksheet that was used in 2.1, “Planning for VMSSI with LGR”, in *The Virtualization Cookbook for IBM z Systems Volume 1: IBM z/VM 6.3,* SG24-8147.

This worksheet is included for your convenience, and organized in the order that you need the data.

It is recommended that you specify all values that apply to make your installation process go more smoothly.

IBM Shopz

If you are ordering z/VM by using Shopz, use Table A-2 to record the values that you use.

Ordering z/VM by using Shopz is described in *The Virtualization Cookbook for IBM z Systems Volume 1: IBM z/VM 6.3,* SG24-8147.

Table A-2 Shopz data

Name	Value	Comment
Starting URL	ibm.com/shopz	
User ID		Customer number. (For IBM employees, it is your intranet user ID and password.)
Password		
Order number		

Hardware Management Console

For more information about how to start a z/VM installation from the Hardware Management Console, see 4.3.1, “Start the z/VM installation” in *The Virtualization Cookbook for IBM z Systems Volume 1: IBM z/VM 6.3,* SG24-8147.

Complete Table A-3 to record the values that you use.

Table A-3 HMC values

Name	Value
HMC host name or URL	
HMC user ID	
HMC password	
FTP source system (if you are installing from FTP)	
z/VM installation directory	

z/VM Installation Planning Panels (INSTPLAN)

For information about the **INSTPLAN** command that is run from the Integrated 3270 Console, see 4.3.2, “Copy a plain z/VM system to DASD” in *The Virtualization Cookbook for IBM z Systems Volume 1: IBM z/VM 6.3,* SG24-8147.

The following information is necessary.

INSTPLAN panels 1 and 2

Complete Table A-4 to record the values that are required in the first two INSTPLAN panels.

Table A-4 INSTPLAN values for first two panels

Name	Value	Comment
Language	☐ AMENG ☐ USENG ☐ KANJI	AMENG (American English), UCENG (uppercase English), or KANJI.
DASD model	☐ 3390 Model-3 ☐ 3390 Model-9	3390 Model-3 or Model-9 (installation to the fixed-block architecture (FBA) disk is not described in this book).
File pool name		VMPSYS (default) is recommended.
System type		SSI. (Non-SSI is not described in this book.)
Non-SSI system name		Used for non-SSI installation only.
Number of members		SSI installation only (usually 2 or 4).
Single system image (SSI) cluster name		SSI installation only.
Automatic configuration		“No” is strongly recommended.

INSTPLAN panel 3

Complete Table A-5 to record the values that are required in the third INSTPLAN panel. The member names become the z/VM system identifiers, and the logical partition (LPAR) names must be the same names as the names on the HMC.

Table A-5 INSTPLAN values for panel 3

Slot	Member name	LPAR name	Comment
1			Member 1 system identifier and LPAR name
2			Member 2 system identifier and LPAR name
3			Member 3 system ID and LPAR name (optional)
4			Member 4 system ID and LPAR name (optional)

INSTPLAN worksheet 3

Complete Table A-6 to record the volume labels and real device addresses that are required on the Installation Volume Definition INSTPLAN panel.

Table A-6 INSTPLAN values worksheet for volume definition

Type	Default label	Chosen label	Address	Comment
COMMON	VMCOM1			Common volume 1
COMMON2	VMCOM2			Common volume 2
RELVOL	630RL1			Release volume 1
RELVOL2	630RL2			Release volume 2
Mem 1 RES	M01R01			Member 1 residence volume
Mem 1 SPOOL	M01S01			Member 1 spool volume
Mem 1 PAGE	M01P01			Member 1 page volume
Mem 1 WORK	M01W01			Member 1 work volume 1
Mem 1 WORK	M01W02			Member 1 work vol 2 (3390-3 only)
Mem 1 WORK	M01W03			Member 1 work vol 3 (3390-3 only)
Mem 2 RES				Member 2 residence volume
Mem 2 SPOOL				Member 2 spool volume
Mem 2 PAGE				Member 2 page volume
Mem 2 WORK				Member 2 work volume 1
Mem 2 WORK				Member 2 work vol 2 (3390-3 only)
Mem 2 WORK				Member 2 work vol 3 (3390-3 only)
Mem 3 RES				Member 3 residence vol (optional)
Mem 3 SPOOL				Member 3 spool volume
Mem 3 PAGE				Member 3 page volume
Mem 3 WORK				Member 3 work volume 1

Type	Default label	Chosen label	Address	Comment
Mem 3 WORK				Member 3 work vol 2 (3390-3 only)
Mem 3 WORK				Member 3 work vol 3 (3390-3 only)
Mem 4 RES				Member 4 residence vol (optional)
Mem 4 SPOOL				Member 4 spool volume
Mem 4 PAGE				Member 4 page volume
Mem 4 WORK				Member 4 work volume 1
Mem 4 WORK				Member 4 work vol 2 (3390-3 only)
Mem 4 WORK				Member 4 work vol 3 (3390-3 only)

INSTPLAN worksheet 4

Complete Table A-7 to record the common volume and channel-to-channel (CTC) addresses that are required in the INSTPLAN panel.

Complete the worksheet in Table A-7 to document your values. For more information about this worksheet, see 4.3.2, “Copy a plain z/VM system to DASD” in *The Virtualization Cookbook for IBM z Systems Volume 1: IBM z/VM 6.3,* SG24-8147.

If only two members are in the SSI, you must specify only two pairs of CTCs (from member 1 to member 2, and vice versa).

Table A-7 INSTPLAN values worksheet for volume definition

Real addresses for the common volume on each member LPAR			
Member 1	**Member 2**	**Member 3**	**Member 4**
CTC device addresses			
From member 1		**From member 2**	
To: member 1	N/A	To: member 1	_____ _____
To: member 2	_____ _____	To: member 2	N/A
To: member 3	_____ _____	To: member 3	_____ _____
To: member 4	_____ _____	To: member 4	_____ _____
From member 3		**From member 4**	
To: member 1	_____ _____	To: member 1	_____ _____
To: member 2	_____ _____	To: member 2	_____ _____
To: member 3	N/A	To: member 3	_____ _____
To: member 4	_____ _____	To: member 4	N/A

z/VM Networking resources

Complete the worksheet in Table A-8 to list the networking resources that are needed when you start the **IPWIZARD** and when you create a virtual switch (VSWITCH) for the Linux virtual machines.

Table A-8 z/VM and networking resources worksheet

Name	Value	Comment
TCP/IP user ID		TCPIP is recommended.
z/VM host name, member 1		
z/VM host name, member 2		
TCP/IP domain name		System domain name that is usually set in DNS.
TCP/IP gateway		The router to and from the local subnet.
DNS server 1		Assigned by the network administrator.
DNS server 2/3		Optional.
Interface name		
OSA starting device number		Start of OSA *triplet* for z/VM TCP/IP stack.
Subnet mask		Assigned by network administrator.
OSA device type		
Maximum transmission unit (MTU) size		Check with network administrator.
Primary OSA device for virtual switch		Specify the first real device number and the next two device numbers will also be used.
Secondary OSA device for virtual switch		Ideally, it needs to be on a different channel-path identifier (CHPID)/OSA card.

z/VM DASD worksheet

Use the worksheet in Table A-9 to document the z/VM DASD that you use.

Table A-9 z/VM DASD blank worksheet

Device number	Label	Type	Notes

Device number	Label	Type	Notes

Linux resources worksheet

Use the worksheet in Table A-10 to document the resources that are associated with the Network File Server (NFS) that are the installation source of the first Linux on z Systems.

Table A-10 Linux NFS server resources blank worksheet

Name	Value	Comment
TCP/IP address		
User/password		
NFS-exported installation directory		

Use the worksheet in Table A-11 to document your Linux on z Systems resources.

Table A-11 Linux resources blank worksheet

Name	Value	Comment
Linux installation password		
Linux root password		
Linux TCP/IP gateway		
Linux TCP/IP broadcast		
Linux DNS server		
Virtual Network Computing (VNC) installation password		

11.8.1 Host names and IP addresses worksheet

Use the worksheet in Table A-12 to document the host names and associated IP addresses and virtual machines that you use.

Table A-12 Host names blank worksheet

Host name	IP address	Virtual machine/ LPAR	Notes

B

Additional material

This appendix refers to additional material that can be downloaded from the Internet.

Locating the web material

The web material that is associated with this book is available on the internet. This web material is the same material that is used for SG24-8147. You can find this material at this web page.

Using the web material

The files that are associated with this book are in a GNU compressed tar file.

The additional web materials that accompany this book are in the following file:

File name	*Description*
SG248147.tgz	Code samples in compressed tar format

Within the tar file, the directory `SG248147/` contains the following subdirectories and files:

```
disclaimer.txt Legal disclaimer
README.txt Description file
lnxmaint\Generic.prm
lnxmaint\Profckd.exec
lnxmaint\Proffba.exec
lnxmaint\Profile.exec
lnxmaint\Redhat.exec
lnxmaint\Rescue.exec
lnxmaint\Rescue.prm
lnxmaint\Swapgen.exec
lnxmaint\Swapgen.helpcms
lnxmaint\Swapgenh.psbin
maintvrm\Operator.profexec
maintvrm\Vmarc.module
maintvrm\Vmcron.exec
maintvrm\Vmlogs.vmarc
maintvrm\Vmserve.vmarc
maintvrm\Vmww2.vmarc

lnxmaint
```

System requirements for downloading the web material

The web material requires the following system configuration:

Hard disk space: 41 KB
Operating system: Linux

Downloading and extracting the web material

This section lists the files that are associated with this book. The following sections are included:

- “z/VM REXX EXECs and XEDIT macros” on page 243
- “Sample files” on page 256

Disclaimer

IBM DOES NOT WARRANT OR REPRESENT THAT THE CODE PROVIDED IS COMPLETE OR UP-TO-DATE. IBM DOES NOT WARRANT, REPRESENT OR IMPLY RELIABILITY, SERVICEABILITY OR FUNCTION OF THE CODE. IBM IS UNDER NO OBLIGATION TO UPDATE CONTENT NOR PROVIDE FURTHER SUPPORT.

ALL CODE IS PROVIDED “AS IS,” WITH NO WARRANTIES OR GUARANTEES WHATSOEVER. IBM EXPRESSLY DISCLAIMS TO THE FULLEST EXTENT PERMITTED BY LAW ALL EXPRESS, IMPLIED, STATUTORY AND OTHER WARRANTIES, GUARANTEES, OR REPRESENTATIONS, INCLUDING, WITHOUT LIMITATION, THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF PROPRIETARY AND INTELLECTUAL PROPERTY RIGHTS. YOU UNDERSTAND AND AGREE THAT YOU USE THESE MATERIALS, INFORMATION, PRODUCTS, SOFTWARE, PROGRAMS, AND SERVICES, AT YOUR OWN DISCRETION AND RISK AND THAT YOU WILL BE SOLELY RESPONSIBLE FOR ANY DAMAGES THAT MAY RESULT, INCLUDING LOSS OF DATA OR DAMAGE TO YOUR COMPUTER SYSTEM.

IN NO EVENT WILL IBM BE LIABLE TO ANY PARTY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY OR CONSEQUENTIAL DAMAGES OF ANY TYPE WHATSOEVER RELATED TO OR ARISING FROM USE OF THE CODE FOUND HEREIN, WITHOUT LIMITATION, ANY LOST PROFITS, BUSINESS INTERRUPTION, LOST SAVINGS, LOSS OF PROGRAMS OR OTHER DATA, EVEN IF IBM IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THIS EXCLUSION AND WAIVER OF LIABILITY APPLIES TO ALL CAUSES OF ACTION, WHETHER BASED ON CONTRACT, WARRANTY, TORT OR ANY OTHER LEGAL THEORIES.

z/VM REXX EXECs and XEDIT macros

This section lists all of the z/VM execs included in the associated tar file:

- CPFORMAT EXEC
- SSICMD EXEC
- PROFILE EXEC for Linux virtual machines

CPFORMAT EXEC

The following code is for the EXEC that formats multiple DASD by using **CPFMTXA**. It is described in 4.12 “Enabling basic system automation” in *The Virtualization Cookbook for IBM z Systems Volume 1: IBM z/VM 6.3,* SG24-8147.

```
/*************************************************************/
/*                                                           */
/*  This program is provided on an "AS IS" basis, without    */
/*  warranties or conditions of any kind, either express or  */
/*  implied including, without limitation, any warranties    */
/*  or conditions of title, non-infringement,                */
/*  merchantability or fitness for a particular purpose.     */
/*  Neither recipient nor any contributors shall have any    */
/*  liability for any direct, indirect, incidental,          */
/*  special, exemplary, or consequential damages (including  */
/*  without limitation lost profits), however caused and on  */
/*  any theory of liability, whether in contract, strict     */
```

```
/*  liability, or tort (including negligence or otherwise)   */
/*  arising in any way out of the use or distribution of     */
/*  the program or the exercise of any rights granted        */
/*  hereunder, even if advised of the possibility of such    */
/*  damages.                                                 */
/*                                                           */
/*************************************************************/
/*                                                           */
/* Purpose:                                                  */
/*   CP format one, a range or multiple ranges of DASD.      */
/*   and label these DASDs.                                  */
/*                                                           */
/* Inputs:                                                   */
/*   dasds - address(es) of the DASD to format.              */
/*   type  - type of formatting to be done: PERM, PAGE, SPOL */
/*           or TEMP.                                        */
/*                                                           */
/* Output:                                                   */
/*   Virtual DASD that is CP formatted and labeled.          */
/*                                                           */
/* Return codes:                                             */
/*   0 - success                                             */
/*   1 - help was asked for or given                         */
/*   2 - user did not respond Y to confirm formatting        */
/*   3 - DASD (minidisk) range is not valid                  */
/*   4 - at least one DASD (minidisk) is reserved to MAINT   */
/*                                                           */
/*                                                           */
/*************************************************************/
  Address COMMAND
  firstchar = 'J'
  Arg dasds 'AS ' type .
  If dasds = '' | dasds = '?' Then Call help
  labelPrefix = firstchar || getLabelPrefix(type)
  numDasd = parseDasd(dasds)
  answer = areYouSure(type)
  If answer = 'Y' Then Do
     /* the user is sure */
     formatted = ''
     retVal = doFormat(labelPrefix numDasd type)
     Call doReport retVal
     End
  Else retVal = 2
  Exit retVal

/*+-------------------------------------------------------------------+*/
help:
  Procedure Expose firstchar
/*+-------------------------------------------------------------------+*/
  Parse Source . . fn .
  Say
  Say 'Synopsis:'
  Say
  Say '  Format and label DASD as page, perm, spool or temp disk space'
  Say '  The label written to each DASD is' firstchar || '<t><xxxx> where:'
```

```
  Say '    <t> is type - P (page), M (perm), S (spool) or T (Temp disk)'
  Say '    <xxxx> is the 4 digit address'
  Say
  Say 'Syntax is:'
  Say "                <---------------<             "
  Say "   >>--CPFORMAT--.-vdev--------.--AS---.-PERM-.--------><"
  Say "                 '-vdev1-vdev2-'       '-PAGE-'"
  Say "                                       '-SPOL-'"
  Say "                                       '-TEMP-'"
  Say
  Exit 1

/*+---------------------------------------------------------------+*/
areYouSure:
  Procedure
/*| Warn the user of possible data loss and ask if it is okay to  |*/
/*|  format the DASD.                                             |*/
/*|  parm 1: format type for the virtual DASD                     |*/
/*|  retVal: first character of response. continue if 'Y'.        |*/
/*+---------------------------------------------------------------+*/
  Arg type
  Say
  Say 'WARNING - this will destroy data!'
  Say 'Are you sure you want to format the DASD as' type 'space (y/n)?'
  Pull answer .
  Return  'LEFT'(answer,1) /* from areYouSure */

/*+---------------------------------------------------------------+*/
getLabelPrefix:
  Procedure expose firstchar
/*| Return the second character of the virtual DASD label         |*/
/*|  parm 1: format type for the virtual DASD                     |*/
/*+---------------------------------------------------------------+*/
  Arg type .
  firstchar. = 0
  firstchar.PERM = 'M'
  firstchar.PAGE = 'P'
  firstchar.SPOL = 'S'
  firstchar.TEMP = 'T'
  If firstchar.type = 0 Then Do
  /* Incorrect formatting type specified. Provide help and quit. */
     Say 'Error: "AS" must be present, type must be PERM, PAGE, SPOL or TEMP'
     Call help
     End
  Return firstchar.type

/*+---------------------------------------------------------------+*/
parseDASD:
  Procedure Expose dasdList.
/*| parse all dasd into an array verifying all are attached       |*/
/*|  parm 1: dasds - the list of dasd passed in                   |*/
/*|  retVal: number of DASD in dasdList                           |*/
/*+---------------------------------------------------------------+*/
  Arg dasds
  numDasd = 0
```

```
  dropheader = ''
  Say
  Say 'Format the following DASD:'
  Do While dasds <> ''
     Parse Upper Var dasds dasd dasds
     dashPos = 'POS'('-',dasd)
     If dashPos = 0 Then Do
     /* There is a singleton DASD specified. */
     /* start and end of range are the same. */
        startrange = dasd
        endrange = dasd
        End
     /* process the range of DASD */
     Else Parse Var dasd startrange  '-' endrange
     Do i = 'X2D'(startrange) To 'X2D'(endrange)
        numDasd = numDasd + 1
        dasdList.numDasd = 'D2X'(i)
        'PIPE CP QUERY MDISK' dasdList.numDasd 'LOCATION',
        dropheader,
        '|CONS'
        If rc <> 0 Then Do
           Say 'Return code from QUERY MDISK =' rc
           /* If RC=40, then HCPxxx40E has been issued and msg below */
           If rc = 40 Then Say 'DASD' dasdList.numDasd 'is not attached.'
           Exit 3
           End
        Call checkReserved(dasdList.numDasd)
        dropheader = '|DROP 1'
        End
     End
  Return numDasd /* from parseDasd */

/*+---------------------------------------------------------------------+*/
doFormat:
  Procedure Expose dasdList. formatted
/*| Format all DASD specified using CPFMTXA                             |*/
/*|  parm 1: labelPrefix - the two character label prefix               |*/
/*|  parm 2: numDasd - number of DASD in the array dasdList             |*/
/*|  parm 3: type - the type of DASD format                             |*/
/*|  retVal: 0 = success                                                |*/
/*+---------------------------------------------------------------------+*/
  Arg labelPrefix numDasd type
  /* Save the current settings for MORE */
  Parse Value 'DIAG'('08','CP QUERY TERM') With ' MORE' morevalues ','
  'CP TERM MORE 1 1' /* Make MORE brief */

  /* Save system identifier and SSI name */
  'PIPE CP QUERY USERID | SPEC W3 | VAR systemID'
  'PIPE CP QUERY SSI | LOCATE /SSI Name/ | SPEC W3 | VAR SSIname'
  If (SSIname = "SSINAME") Then /* variable not set */
    inSSI = 'no'
  Else
    inSSI = 'yes'

  /* Iterate through all DASD in list */
```

```
  Do i = 1 to numDasd
     label = labelPrefix  || 'RIGHT'(dasdList.i,4,'0')
     retVal = formatOne(dasdList.i type label)
     If retVal <> 0 Then Do
        Say 'Error from CPFMTXA on DASD' label 'rc =' retVal
        Leave /* error - abort this format */
        End

     /* add owner info for CP owned devices */
     If (type != 'PERM') Then /* CP owned => owner info is needed */
       If (inSSI = 'yes') Then /* add owner info */
         call addOwnerInfo(dasdList.i label SSIname systemID)
       Else
         call addOwnerInfo(dasdList.i label "NOSSI" systemID)
     formatted = formatted label
     End /* Do i = */
  'CP TERM MORE' morevalues
  Return retVal /* from doFormat */

/*+---------------------------------------------------------------+*/
checkReserved:
  Procedure
/*| Try copying an already formatted DASD Then relabelling it     |*/
/*|  parm 1: dasd - the virtual address of the DASD               |*/
/*+---------------------------------------------------------------+*/
  Arg dasd
  /* Create a list of reserved virtual DASD addresses. */
  /* Ensure that a system minidisk is not formatted.   */
  resvd = '122 123 124 190 191 193 19D 19E 2CC 401 402 990 CF1 CF3 CFD'
  If 'POS'(resvd,dasd) <> 0 Then Do
     /* MAINT minidisk - ABORT! */
     Say 'Minidisk' dasd 'is a reserved MAINT minidisk'
     Say 'This must be formatted manually using a different vaddr.'
     Exit 4
     End /* If dasd is reserved */
  Return /* from checkReserved */

/*+---------------------------------------------------------------+*/
doReport:
  Procedure Expose dasds formatted
/*| Report on the newly labelled DASD                             |*/
/*|  parm 1: formatSuccess - 0=all is well, non-0= a format failed|*/
/*|  retVal: 0 = success                                          |*/
/*+---------------------------------------------------------------+*/
  Arg formatSuccess
  If formatSuccess <> 0 Then
    Say 'Error was encountered! retVal from CPFMTXA =' formatSuccess
  If formatted = '' Then
    Say 'No DASD were successfully formatted'
  Else
    Say 'DASD successfully formatted:' formatted
  'CP DETACH' dasds
  'CP ATTACH' dasds '*'
  Say
  Say 'DASD status after:'
```

```
  'CP QUERY MDISK' dasds 'LOCATION'
  Return 0 /* from doReport */

/*+---------------------------------------------------------------+*/
formatOne:
  Procedure
/*| Format a DASD via DDR                                         |*/
/*|  parm 1: disk - the vaddr to be formatted                     |*/
/*|  parm 2: type - PERM, PAGE, SPOL or TEMP                      |*/
/*|  parm 3: label - the six character label                      |*/
/*+---------------------------------------------------------------+*/
  Arg disk type label
  Queue 'FORMAT'
  Queue disk
  Queue '0 END'
  Queue label
  Queue 'YES'
  Queue type '0 END'
  Queue 'END'
  'EXEC CPFMTXA'
  retVal = rc
  Return retVal /* from formatOne */

/*+---------------------------------------------------------------+*/
AddOwnerInfo:
  Procedure
/*| Tag PAGE, SPOL and TDSK volumes with SSI                      |*/
/*|  parm 1: disk - the vaddr to be formatted                     |*/
/*|  parm 2: type - PERM, PAGE, SPOL or TEMP                      |*/
/*|  parm 3: label - the six character label                      |*/
/*+---------------------------------------------------------------+*/
  Arg disk label SSIname systemID
  Queue 'OWNER'
  Queue disk
  Queue label
  Queue SSIname
  Queue systemID
  'EXEC CPFMTXA'
  retVal = rc
  Return retVal /* from addOwnerInfo */
```

SSICMD EXEC

The following code is for the EXEC that issues control program (CP) commands on all joined members of a single system image (SSI) cluster. It is recommended to reside on the `MAINT 191` disk.

```
/*************************************************************/
/*                                                           */
/*  This program is provided on an "AS IS" basis, without    */
/*  warranties or conditions of any kind, either express or  */
/*  implied including, without limitation, any warranties    */
/*  or conditions of title, non-infringement,                */
/*  merchantability or fitness for a particular purpose.     */
/*  Neither recipient nor any contributors shall have any    */
```

```
/*  liability for any direct, indirect, incidental,          */
/*  special, exemplary, or consequential damages (including  */
/*  without limitation lost profits), however caused and on  */
/*  any theory of liability, whether in contract, strict     */
/*  liability, or tort (including negligence or otherwise)   */
/*  arising in any way out of the use or distribution of     */
/*  the program or the exercise of any rights granted        */
/*  hereunder, even if advised of the possibility of such    */
/*  damages.                                                 */
/*                                                           */
/*************************************************************/
/*                                                           */
/* Purpose:                                                  */
/*   Issue a command on all members of a cluster using the   */
/*   response from QUERY SSI to find the member names.       */
/*                                                           */
/* Inputs:                                                   */
/*   cmd - the CP command to issue on each member.           */
/*                                                           */
/* Output:                                                   */
/*   The results from issuing the AT command.                */
/*                                                           */
/*                                                           */
/*************************************************************/
  Address COMMAND
/* The command is passed by the caller */
  Arg cmd
  /* Provide help if requested or if no command is specified */
  If cmd = '' | cmd = '?' Then Call Help
  /* Determine the members of the SSI cluster */
  'PIPE CP QUERY SSI',
  '| STEM MSG.',           /* Save the response if error */
  '| XLATE',               /* Make all output upper case */
  '| FRTARGET ALL /SLOT/', /* Just look after 'SLOT' */
  '| LOCATE /JOINED/',     /* JOINED members can do a command */
  '| SPEC W2',             /* Get the member names */
  '| STEM SSI.'            /* Save the member names */
  /* If nonzero return code, show error message and exit */
  If rc <> 0 | ssi.0 = 0 Then Do
     Say 'Error: QUERY SSI return code =' rc
     Say msg.1
     End
  Else Do
  /* Send the command to each member of the SSI cluster */
     Do i = 1 To ssi.0
        Say ssi.i||":"
        'CP AT' ssi.i 'CMD' cmd
        Say
        End
     End
  Exit

help:
  /* Provide syntax information to the user */
  Say 'SSICMD cmd'
```

```
  Say
  Say 'cmd is a command to be issued on each of the members'
  Say '  in the SSI cluster using the AT command.'
  Exit
```

PROFILE EXEC for Linux virtual machines

This section lists the code for the `PROFILE EXEC` that is shared among Linux virtual machines from the `LNXMAINT 192` disk:

```
/* PROFILE EXEC for Linux virtual servers */
'CP SET RUN ON'
'CP SET PF11 RETRIEVE FORWARD'
'CP SET PF12 RETRIEVE'
'ACC 592 C'
'SWAPGEN 300 524288' /* create a 256M VDISK disk swap space */
'SWAPGEN 301 1048576' /* create a 512M VDISK disk swap space */
'PIPE CP QUERY' userid() '| var user'
parse value user with id . dsc .
if (dsc = 'DSC') then /* user is disconnected */
  'CP IPL 100'
else  /* user is interactive -> prompt */
 do
  say 'Do you want to IPL Linux from minidisk 100? y/n'
  parse upper pull answer .
  if (answer = 'Y') then 'CP IPL 100'
 end
```

REDHAT EXEC

This section lists the code for the `REDHAT EXEC` that starts a RHEL installation. It is recommended to be on the `LNXMAINT 192` disk:

```
/*************************************************************/
/* Punch a RHEL install system to reader and IPL it */
/*************************************************************/
'CL RDR'
'PURGE RDR ALL'
'SPOOL PUNCH * RDR'
'PUNCH KERNEL  IMG * (NOH'
'PUNCH GENERIC PRM * (NOH'
'PUNCH INITRD  IMG * (NOH'
'CH RDR ALL KEEP NOHOLD'
'I 00C'
```

SWAPGEN EXEC

The following code is for the EXEC that creates Linux swap spaces from z/VM VDISKs:

```
/*****************************************************************
* Program: SWAPGEN EXEC
*
* Original Author: Dave Jones (djones@sinenomine.net)
*
* Description/Purpose:
*
* Generate VDISK swap for Linux on System z guest virtual
* machines
*
* Syntax:
*
* Issue: SWAPGEN ? for syntax etc.
*
* Version History:
...
*/

   address command

   arg vdev blks . '(' options ')'

   debug = 0                                    /* Default to quiet */
   fba = 0                                     /* No FBA option yet */
   reuse = 0                                  /* No reuse option yet */
   do while options <> ''                       /* Parse the options */
      parse var options option options             /* Get an option */
      select
         when option = 'DIAG' then fba = 0        /* Use DIAG driver */
         when option = 'FBA' then fba = 1          /* Use FBA driver */
         when option = 'REUSE' then reuse = 1          /* Reuse DASD */
         when option = 'DEBUG' then debug = 1    /* Wants debug chat */
         when option = 'VERSION' then signal Version /* version query*/
         otherwise
         say 'Invalid option "'option'"'             /* Else unknown */
      end
   end

   minblks = 40 - 8 * fba  /* Minimum number of blocks that can work */

   if reuse = 1 then do
      parse value diagrc(8, 'Q V 'vdev) ,     /* Get blocks from ... */
        with rc . 17 msg                   /* ... actual device size */
      if rc <> 0 then signal BadDev
      parse var msg . . . . . newblks .
      if blks = '' then blks = newblks   /* Default to detected size */
      if blks <> newblks then signal WrongBlks    /* Mismatch, error */
   end

   if vdev = '?' then signal Help             /* Wants Help, give it */
   if vdev = '' then signal NoVdev                 /* Missing, error */
   if blks = '' then signal NoBlks                 /* Missing, error */
```

```
if datatype(blks, 'W') = 0 | blks < minblks then /* Bad/too small */
signal BadBlks                                        /* So error */
if datatype(vdev, 'X') = 0 | length(vdev) > 4 then     /* Invalid */
signal BadVdev                                        /* So error */

if fba then do    /* If FBA driver, make sure we have the package */
   'NUCEXT RXDASD'                             /* Already got it?? */
   if rc <> 0 then 'NUCXLOAD RXDASD'        /* No, try to load it */
   if debug then say 'SWAPGEN: Loading RXDASD got rc=' rc
   if rc <> 0 then signal NoRXDASD       /* That failed, so error */
end

if reuse = 0 then do
   call diag 8, 'DETACH' vdev       /* DETACH any existing device */
   parse value diagrc(8, 'DEFINE VFB-512 AS' vdev 'BLK' blks) ,
     with rc . 17 msg '15'x                  /* Define the V-DISK */
   if debug then say 'SWAPGEN: DEFINE VDEV got rc=' rc
   if rc <> 0 then signal BadDefine      /* That failed, so error */
end
call csl 'DMSGETFM rc reacode  fm'        /* find a free filemode */
if debug then say 'SWAPGEN: Got filemode' fm 'from DMSGETFM'
if rc <> 0 then signal NoFreeModes /* Weren't any, strange, error */

if fba then do
   pages = trunc((blks * 512)/4096) - 1              /* FBA case */
   writeit = 'stem swap.'                           /* Pipe stage */
   if debug then say 'SWAPGEN: Computed' pages 'for FBA disk'
end
else do                 /* Not FBA, we must FORMAT and RESERVE it */
   'MAKEBUF'         /* Guard stack contents if something's there */
   buf = rc    /* Remember buffer number so we drop the right one */
   if debug then say 'SWAPGEN: Acquired buffer' buf ,
     'before non-FBA format.'
   writeit = 'specs number 1 1-* next' ,
     '| mdskupdate LINUX SWAP' fm 'F 512'         /* Pipes stages */
   queue '1'                       /* Yes to the format? question */
   queue 'LXSWAP'                             /* Disk volume name */
   queue '1'                       /* Yes to the reserve question */
   'PIPE (name SWPFORMAT)' ,       /* FORMAT and RESERVE the disk */
     '| cms FORMAT' vdev fm '(BLKSIZE 512 NOERASE' ,    /* FORMAT */
     '| var rs1' ,                      /* Remember how that went */
     '| hole' ,                         /* And otherwise pitch it */
     '| cms RESERVE LINUX SWAP' fm ,                    /* Do it */
     '| var rs2' ,                      /* Remember how that went */
     '| hole' ,                         /* And otherwise pitch it */
     '| state LINUX SWAP' fm  , /* Look at the reserved swap file */
     '| var reserveok' ,                /* Keep that information */
     '| specs w6 1'          ,  /* Word 6 is the number of blocks */
     '| specs w1 1' ,                   /* Calculate it modulo 8 */
     'a: word 1 .' ,                   /* Get the token we want */
     '   set #0:=a%8-2' ,        /* Calculate it modulo 8 minus 2 */
     '   print #0 20' ,                              /* Write it */
     '| specs w2 1' ,      /* Just get the number of usable pages */
     '| var pages'                           /* And remember that */
   if debug then say 'SWAPGEN: Formatted' pages 'pages on disk' ,
```

```
        fm 'in PIPE'
      'DROPBUF' buf          /* Not nice to leave trash lying around */
      if debug then say 'SWAPGEN: Dropped buffer' buf
   end

   if debug then say 'SWAPGEN: About to write non-FBA swap signature'

/* Must use separate Pipe to write since mdskupdate commits to 0 */
   'PIPE (name SWPWrite)' ,
     '| var pages' ,                        /* Get number of pages */
     '| specs pad 00 w1 d2c 1.4 right' ,              /* Format it */
     '| append strliteral x'c2x(copies('00'x, 4086-1033+1) || ,
     '53574150535041434532'x) ,            /* "SWAPSPACE2" in ASCII */
     '| join' ,                     /* Build that into a nice chunk */
     '| preface strliteral x'c2x(copies('00'x, 1027)'01'x) ,   /* 0s */
     '| join' ,                     /* Build that into a nice chunk */
     '| deblock 512',                         /* Break into records */
     '|' writeit   /* And write to disk or variable, per driver type */
   if debug then say 'SWAPGEN: Wrote non-FBA swap signature with rc=' rc
   if rc <> 0 then signal BadWrite

/* If FBA, we have the values, need to use RXDASD to write them */
   if fba then do i = 1 to swap.0     /* If FBA, we didn't write yet */
      if debug then say 'SWAPGEN: About to write FBA signature' i
      rc = DASD('WRITED', vdev, i-1, swap.i)           /* Write one */
      if debug then say 'SWAPGEN: Wrote FBA signature' i 'with rc='rc
      if rc <> 0 then signal BadWrite           /* Failed, so error */
   end

   if fba then type ='FBA'
   else type = 'DIAG'
   say type 'swap disk defined at virtual address' vdev , /* Success! */
     '('pages-1' 4K pages of swap space)'
   call Quit 0

Quit:
   arg rc
   if rc <> 0 then say 'No Swap disk was created.'
   exit rc

NoVdev:              /* User didn't give us a virtual device address */
   say 'A virtual device address must be specified!'
   signal Help

NoBlks:                    /* User didn't give us a number of blocks */
   say 'Number of blocks must be specified!'
   signal Help

NoFreeModes:                      /* No free disk modes can be found */
   say 'No free disk modes are available!'
   say 'Please release a minidisk and try again.'
   call Quit 1      /* They invoked it correctly, so don't show help */

BadDev:                           /* REUSE tried to use bad device */
   say 'The device at 'vdev' cannot be used:'
```

```
   say msg
   call Quit 24

BadBlks:                 /* User gave us an invalid number of blocks */
   say 'Invalid number of blocks "'blks'" specified; must be'
   say 'at least 'minblks' 512-byte blocks.'
   call Quit 24

WrongBlks:               /* Supplied number of blocks does not match */
   say 'REUSE requested with' blks ,
     'and existing disk block count is' newblks'.'
   call Quit 24

BadVdev:           /* User gave us an invalid virtual device address */
   say 'Invalid virtual device address "'vdev'" specified;'
   say 'must be a 1- to 4-digit hexadecimal value.'
   call Quit 24

NoRXDASD:                  /* We don't have the required FBA utility */
   say 'Unable to NUXCLOAD RXDASD MODULE; this is available from:'
   say ' http://www.vm.ibm.com/download/packages'
   call Quit rc

BadDefine:                              /* Error DEFINE-ing the VDISK */
   say 'Error' rc 'from CP DEFINE VFB-512 AS' vdev 'BLK' blks':'
   say msg                                  /* Display error from CP */
   call Quit rc

BadFBA:                          /* Error writing FBA block on disk */
   say 'Error' rc 'from RXDASD'
   call Quit rc

BadWrite:                       /* Error on FORMAT or RESERVE steps */
   select                         /* Figure out where it went wrong */
      when symbol('RESERVEOK') <> 'VAR' then do
         say 'Error' rc 'from CMS RESERVE LINUX SWAP' fm':'
         say rs2
      end
      when symbol('RS2') <> 'VAR' then do
         say 'Error' rc 'from CMS FORMAT' vaddr fm '(BLKSIZE 512:'
         say rs1
      end
      otherwise
      say 'Error' rc 'calculating swap size, contact support'
   end
   call Quit rc

Help:
   parse source . . fn .

   say 'Syntax is:'
   say ''
   say fn 'vdev #blocks <( <options> <)> >'
   say '   or'
   say fn 'vdev ( REUSE <options> <)>'
```

```
   say ''
   say 'where:'
   say ''
   say 'vdev     -- is a virtual device address'
   say '#blocks  -- is a decimal number of 512-byte blocks;'
   say '            minimum 24 (FBA) or 32 (DIAG)'
   say ''
   say 'Options are:'
   say 'DIAG     -- (Default) Use DIAG I/O (requires Linux DIAG driver)'
   say 'FBA      -- use FBA driver instead of DIAG; requires RXDASD'
   say '            package, downloadable from the IBM VM download'
   say '            page at: http://www.vm.ibm.com/download/packages'
   say 'REUSE    -- use existing device at vdev.  WARNING: This will'
   say '            destroy any data on device vdev.  The #blocks'
   say '            parameter may be omitted; the whole device will'
   say '            be used in that case.'
   say 'VERSION  -- display current version number string and date'
   say '            of last module update.'
   say 'DEBUG    -- display progress messages and debugging'
   say '            information about the program logic. '
   say ''
   say fn 'will DETACH any existing virtual device at that address,'
   say 'DEFINE a new VDISK, format it, and write the Linux swap'
   say 'signature on the disk so Linux will recognize it.'
   say ''
   say 'If using FBA mode, SWAPGEN prepares the whole device:'
   say '    /dev/dasdb or /dev/dasd/0151/device'
   say 'so the whole device must then specified in the Linux fstab.'
   say ''
   say 'If using DIAG mode, because the V-DISK is CMS FORMATted,'
   say 'SWAPGEN prepares the partition:'
   say '    /dev/dasdb1 or /dev/dasd/0151/part1'
   say 'so the partition must be specified in the fstab on Linux.'
   call Quit 1

Version:
   parse source . . fn .

/* These variables should be updated with each release */
   version = 'SNA120601'               /* Release string: SNAyymmvv */
   last_update = '2012-06-20 (yyyy-mm-dd)'       /* Last update date */

   say 'SWAPGEN: Version' version', last updated:' last_update'.'
   call Quit 4
```

Sample files

This section lists sample files that are described in the book.

SAMPLE GENERIC PRM files

This following is a sample RHEL 7.1 configuration file for ECKD/EDEV:

```
ro ramdisk_size=40000 cio_ignore=all,!condev
                  ip=<n.n.n.n>::<n.n.n.n>:<n>:<HOSTNAME>:enccw0.0.0600:none
                  rd.znet=qeth,0.0.0600,0.0.0601,0.0.0602,layer2=1
                  nameserver=<n.n.n.n> nameserver=<n.n.n.n>
                  inst.repo=ftp://lydiap:new2day@n.n.n.n//<RHEL Install Tree>
                  rd.dasd=<DASD> rd.dasd=<DASD2>
                  vnc vncpassword=12345678
```

This following is a sample RHEL 7.1 configuration file for FCP using Kickstart:

```
ro ramdisk_size=40000 cio_ignore=all,!condev
ip=<n.n.n.n>::<n.n.n.n>:<n>:<HOSTNAME>:enccw0.0.0600:none
rd.znet=qeth,0.0.0600,0.0.0601,0.0.0602,layer2=1
nameserver=<n.n.n.n> nameserver=<n.n.n.n>
rd.zfcp=<FCP1> rd.zfcp=<FCP2>
inst.repo=ftp://<n.n.n.n>/<RHEL Install Tree>
ks=ftp://<n.n.n.n>/<Path to kickstart file>
inst.cmdline
```

Related publications

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

IBM Redbooks

The following IBM Redbooks publications provide more information about the topic in this document. Several publications that are referenced in this list might be available in softcopy only:

- *The Virtualization Cookbook for IBM z Systems Volume 1: IBM z/VM 6.3,* SG24-8147
- *The Virtualization Cookbook for IBM z Systems Volume 3: SUSE Linux Enterprise Server 12,* SG24-8890
- *Linux on IBM eServer zSeries and S/390: Performance Toolkit for VM*, SG24-6059
- *z/VM and Linux on IBM System z*, SG24-7492
- *Linux on IBM eServer zSeries and S/390: Application Development*, SG24-6807
- *IBM Lotus Domino 6.5 for Linux on zSeries Implementation*, SG24-7021
- *Printing with Linux on zSeries Using CUPS and Samba*, REDP-3864
- *IBM z Systems Connectivity Handbook*, SG24-5444
- *Deploying a Cloud on IBM System z*, REDP-4711
- *Installing Oracle 11gR2 RAC on Linux on System z*, REDP-4788
- *Linux on IBM System z: Performance Measurement and Tuning*, SG24-6926
- *Fibre Channel Protocol for Linux and z/VM on IBM System z*, SG24-7266
- *Security for Linux on System z*, SG24-7728
- *Advanced Networking Concepts Applied Using Linux on IBM System z*, SG24-7995
- *Set up Linux on IBM System z for Production*, SG24-8137
- *Practical Migration from x86 to Linux on IBM System z*, SG24-8217
- *End-to-End High Availability Solution for System z from a Linux Perspective*, SG24-8233
- *Security for Linux on System z: Securing Your Network*, TIPS0981
- *Linux on System z: An Ideal Platform to Migrate Your IT Workload*, TIPS1166
- *Linux on IBM eServer zSeries and S/390: Performance Toolkit for VM*, SG24-6059
- *Printing with Linux on zSeries Using CUPS and Samba*, REDP-3864

You can search for, view, download, or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications

The following publications are also relevant as further information sources:

- z/VM online documentation:

  http://www.vm.ibm.com/library/

  Under the `z/VM V6.3 PDF Files` link, the following books are useful:

  - *z/VM CP Messages and Codes*
  - *z/VM TCP/IP Messages and Codes*
  - *z/VM CP Commands and Utilities Reference*
  - *z/VM CP Planning and Administration*
  - *z/VM Getting Started with Linux on System z*
  - *z/VM TCP/IP Planning and Customization*
  - *z/VM Performance Toolkit Guide*, SC24-6156
  - *z/VM Performance Toolkit Reference*, SC24-6157

  Under the `Program Directories` link, the following books are useful:

  - *Performance Toolkit for VM*
  - *DirMaint*
  - *RACF Security Server for z/VM*
- *z/VM Performance Toolkit Guide*, SC24-6156
- *z/VM Performance Toolkit Reference*, SC24-6157
- *Getting Started With Linux on System z*, SC24-6096
- *Device Drivers, Features, and Commands on Red Hat Enterprise Linux 7.1*, SC34-2710
- *Device Drivers, Features, and Commands on SUSE Linux Enterprise Server 12*, SC34-2745
- *The Program Directory for Performance Toolkit for VM*, GI10-0785:

  http://www.vm.ibm.com/progdir/6vmptk30.pdf

Online resources

The following websites are also relevant as further information sources:

- Linux for zSeries and S/390 portal:

  http://linuxvm.org
- IBMVM list server:

  http://listserv.uark.edu/archives/ibmvm.html
- Linux-390 list server:

  http://www2.marist.edu/htbin/wlvindex?linux-390
- *Documentation for z Systems Linux Development stream:*

  - http://www.ibm.com/developerworks/linux/linux390/documentation_red_hat.html
  - http://www.ibm.com/developerworks/linux/linux390/documentation_suse.html
- Red Hat Enterprise Linux Server no-charge evaluation download for IBM z Systems:

  http://www.redhat.com/en/technologies/linux-platforms/enterprise-linux

- Kickstart:

  https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Installation_Guide/sect-kickstart-howto.html
- z/VM publications:

  http://www.vm.ibm.com/pubs
- z/VM performance tips:

  http://www.vm.ibm.com/perf/tips
- z/VM VDISK for Linux swap performance tips:

  http://www.vm.ibm.com/perf/tips/lxswpvdk.html
- z/VM TCP/IP planning, customization, and reference:

  http://www.vm.ibm.com/related/tcpip/tcp-pubs.html
- z/VM TCP/IP cryptographic security:

  http://www.vm.ibm.com/related/tcpip/vmsslinf.html
- z/VM user's guides and command references (XEDIT, Conversational Monitor System (CMS), and others):

  http://www.vm.ibm.com/library/zvmpdf.html
- XEDIT for VM/SystemProduct R3 (Historical reference):

  http://ukcc.uky.edu/ukccinfo/391/xeditref.html
- RHEL documentation website:

  http://www.ibm.com/developerworks/linux/linux390/documentation_red_hat.html

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Redbooks

The Virtualization Cookbook for IBM Z Volume 2: Red Hat

SG24-8303-01

ISBN 0738460060

IBM

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

SG24-8303-01

ISBN 0738460060

Printed in U.S.A.