z/VM

Getting Started with Linux on System z

version 6 release 1

SC24-6194-00

z/VM

Getting Started with Linux on System z

version 6 release 1

SC24-6194-00

Note:
Before using this information and the product it supports, read the information under "Notices" on page 143.

This edition applies to version 6, release 1, modification 0 of IBM z/VM, (product number 5741-A07) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC24-6096-03.

Contents

About this document

This document describes how to configure and use z/VM® functions and facilities for Linux® servers running on the IBM® System z® platform (hereafter referred to as the *mainframe*). The document provides requirements and guidelines to implement during z/VM installation, but primarily assumes you have installed z/VM and are ready to deploy Linux in virtual machines.

Early sections acquaint you with z/VM and take you from the point after z/VM installation to the point where you are ready to install your first Linux server. At that point you must turn to the installation documentation provided by your Linux distributor. Following the deployment of your first Linux server, you can replicate or clone additional servers.

When you finish the early sections, you will have two or more Linux servers running on z/VM with TCP/IP connections to the network. Subsequently, you can turn to vendor-supplied documentation to install applications on Linux. Later sections cover operations, administration, performance, and other day-to-day bare essentials.

Intended audience

This document is designed to help anyone who does system programming, administration, or operation, but has limited knowledge of z/VM and wants to get started deploying Linux servers on z/VM. Before you begin, you must:

- Understand mainframe hardware concepts, such as logical partitions (LPARs) and I/O
- Know and have used the Linux operating system
- Know and have used TCP/IP.

The environment for your z/VM system environment is assumed to include:

- A mainframe with an OSA-Express device
- z/VM version 6 release 1
- Directory Maintenance Facility
- Performance Toolkit for VM
- If you do not have an external file server for the Linux code, you might need an NFS or FTP server.

Conventions and terminology used in this document

This document is primarily a cookbook; that is, it provides instructions about how to accomplish a task or goal. When required, background concepts are provided to help you understand a key z/VM function or facility. Instructions and background concepts are separated but linked together through cross-references, providing an efficient path through the instructional material. At the beginning of each set of instructions, you will see a **Before you begin** section, which explains what you need to know or to do before you perform the task. Cross-references in the **Before you begin** section take you to the necessary background concepts. Thus, if you already know the necessary concepts, you do not need to read those background topics and can simply follow the instructions.

Though the topics in this document are self-sufficient, you might want to explore a function or facility in detail. Some topics end with a list of documents that you can use to understand a function or facility in detail.

In general, new terms (in *italics*) are defined in the context they are introduced.

Sometimes the manual focuses on the virtual machine functions (the virtual hardware) and other times the complete Linux server system (the virtual machine and the Linux operating system as a whole). When focusing on the virtual machine functions only, the term *virtual machine* or *virtual machine for Linux* is used. The term *Linux virtual server* refers to the complete Linux system (virtual machine hardware and the Linux operating system as a whole) running on z/VM.

Commands and statements that you must type are in **bold** while system responses are in normal font.

Example: In the example, you would type "query processors". The rest of the example is the system response:

```
query processors
PROCESSOR 00 MASTER
PROCESSOR 01 ALTERNATIVE
Ready;
```

Variable information appears in ***bold italics***, which means you must substitute your own values for the variable.

Example: For the command, you would need to supply your own password for the variable ***new_password***.

```
dirm add linmstr like linux pw new_password
```

A vertical ellipsis

```
⋮
```

indicates system responses that have been removed for clarity.

Where to find more information

For z/VM terms used in this document, see *z/VM: Glossary* or use the z/VM HELP Facility (for more information about glossary terms and the z/VM HELP Facility, see "Glossary" on page 147).

For more information about related publications, see "Bibliography" on page 149.

Additional Publications

For white papers, IBM Redbooks® publications, and other useful information about Linux on the mainframe, visit the z/VM resources for Linux on IBM System z Web site at:

http://www.vm.ibm.com/linux/

Publications you might be interested in are:

- *z/VM and Linux on IBM System z: The Virtualization Cookbook for SLES9*, SG24-6695
- *Security on z/VM*, SG24-7471

Links to Other Online Documents

If you are viewing the Adobe® Portable Document Format (PDF) version of this document, it might contain links to other documents. A link to another document is based on the name of the requested PDF file. The name of the PDF file for an IBM document is unique and identifies the edition. The links provided in this document are for the editions (PDF names) that were current when the PDF file for this document was generated. However, newer editions of some documents (with different PDF names) might exist. A link from this document to another document works only when both documents reside in the same directory.

How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or give us any other feedback that you might have.

Use one of the following methods to send us your comments:

1. Send an e-mail to mhvrcfs@us.ibm.com
2. Visit the z/VM reader's comments Web page at www.ibm.com/systems/z/os/zvm/zvmforms/webqs.html
3. Mail the comments to the following address:
   IBM Corporation
   Attention: MHVRCFS Reader Comments
   Department H6MA, Mail Station P181
   2455 South Road
   Poughkeepsie, NY 12601-5400
   U.S.A.
4. Fax the comments to us as follows:
   From the United States and Canada: 1+845+432-9405
   From all other countries: Your international access code +1+845+432-9405

Include the following information:

- Your name and address
- Your e-mail address
- Your telephone or fax number
- The publication title and order number:
  z/VM V6R1 Getting Started with Linux on System z
  SC24-6194-00
- The topic and page number related to your comment
- The text of your comment

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you submit to IBM.

If you have a technical problem

Do not use the feedback methods listed above. Instead, do one of the following:

- Contact your IBM service representative.
- Contact IBM technical support.
- Visit the z/VM support Web page at www.vm.ibm.com/service/
- Visit the IBM mainframes support Web page at www.ibm.com/systems/support/z/

Summary of changes

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the changes. Some program updates might be provided through z/VM service by program temporary fixes (PTFs) for authorized program analysis reports (APARs), which also might be available for some prior releases.

SC24-6194-00, z/VM Version 6 Release 1

This edition supports the general availability of z/VM V6.1. Changes made are:

- A step was added to "Steps for copying the current USER DIRECT file" on page 57.
- Other minor technical and editorial changes.

Chapter 1. About z/VM

This topic is a z/VM primer and covers general VM concepts, such as editing and finding files, required to complete z/VM system tasks.

When you log onto z/VM, you have the functional equivalent of a real computer and its associated devices at your fingertips. This functional equivalent of a computing system is called a *virtual machine*. Virtual machines are not real, but do work like real systems. Everyone in your entire organization can use z/VM to share the resources of a single computer, while at the same time accessing the system as if each is the only user.

Figure 1 shows a stylized representation of a real mainframe computing system. Each real computing system has one or more central processing units (CPUs), storage (memory), peripheral devices for input and output, such as disks, tapes, printers, and displays, and the operator console. The operating system manages all these resources to do work.

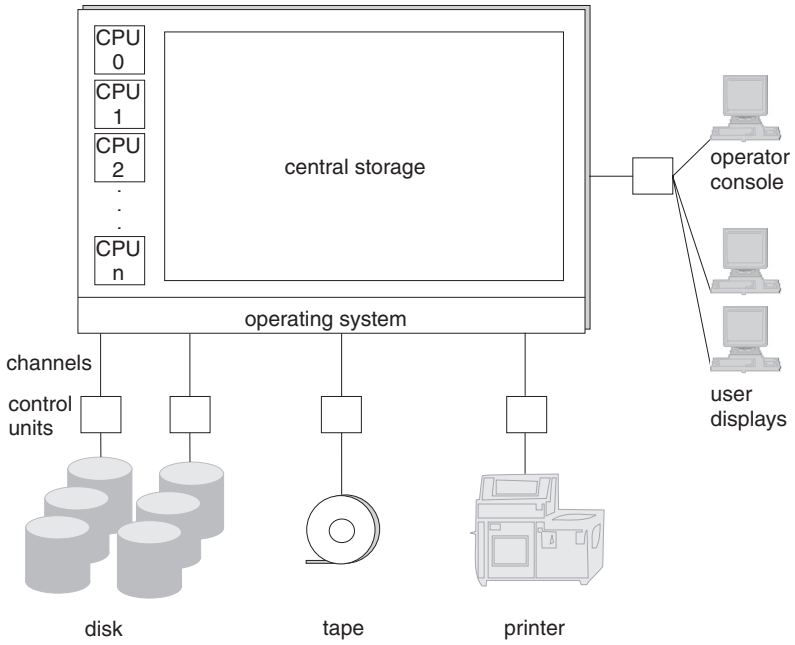

Figure 1. Representation of a mainframe computing system

z/VM virtualizes real computing resources so that each user appears to have a complete mainframe computing system, as shown in Figure 2 on page 2. This means each virtual machine can run its own operating system to manage its virtual resources. It also means you can perform virtual machine tasks as if they were real machine tasks: you can boot (perform an *initial program load* of) an operating system, attach and detach devices, and manage the work of your virtual machine operating system.

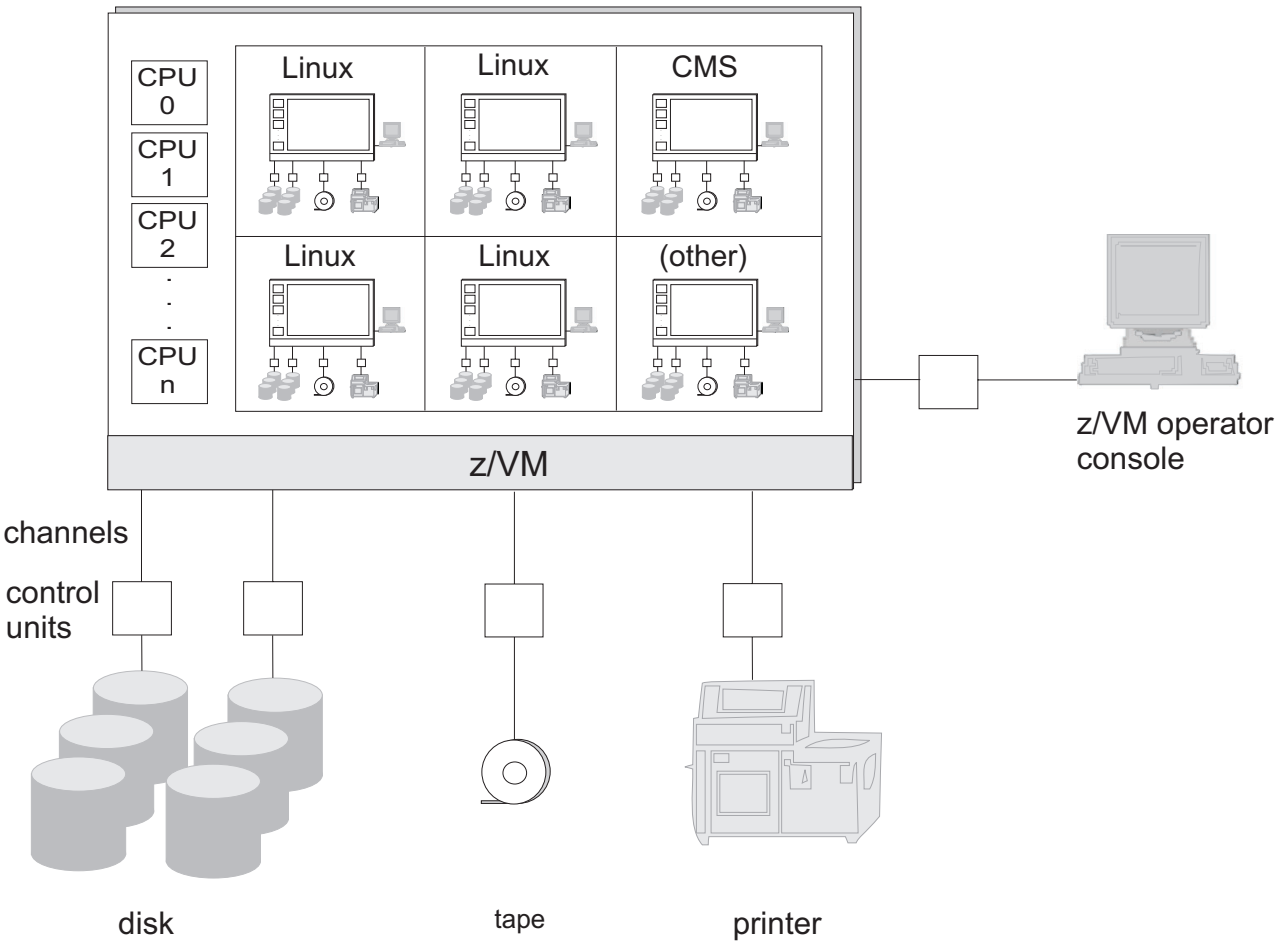

Figure 2. Representation of virtual machines

A virtual machine is directly associated with a z/VM *user ID* or logon identifier. When you log onto z/VM, you have a virtual machine at your disposal and control the virtual machine the way a system operator controls the real hardware.

Some user IDs (virtual machines) are given special privileges to control z/VM and the real machine. For example, the OPERATOR has special privileges allowing control of real machine resources. Another user, usually called MAINT, has special privileges to change z/VM code, apply z/VM maintenance, and add new users. Whether or not users have special privileges, they all perform their tasks through a virtual machine. So, as shown in Figure 2, some virtual machines run Linux, while others run other operating systems, such as the Conversational Monitor System (CMS) (more about CMS in a moment).

The Control Program (CP) is the component of z/VM that manages the resources of a single computer so that multiple computing systems (virtual machines) appear to exist. Think of CP as a supervisor (or *hypervisor*) program running in a layer between the hardware and virtual machines. When you are working in the CP environment, you are provided with CPU (central processing unit) functions, input and output devices, and storage. Through CP, each virtual machine can run its own operating system, such as Linux, z/OS®, or z/VM itself.

Operating systems running in virtual machines are often called *guests*. Other terms and phrases you might encounter are:

- *Running first level*: running directly on the hardware, which is what z/VM does.
- *Running second level* or *running under VM* or *running on (top of) VM*: running as a guest.

During its time slice, a guest actually runs on the real machine. The hardware microcode handles most guest program instructions and CP takes control only when necessary, which maintains good performance.

The Conversational Monitor System (CMS) is the interactive component of z/VM. CMS is a single-user operating system that runs in a virtual machine. Typically, each directory entry (user definition) has a statement that loads the CMS operating system at logon time (see "The user directory" on page 8). CMS is not only an end-user interactive component, but a home for running system utilities and tasks, such as TCP/IP and system management functions. At the z/VM level, systems personnel use CMS to manage z/VM itself and guests. At the guest level, you can use CMS to define resources for your virtual machine, so loading CMS is useful even if you plan subsequently to load Linux into the same virtual machine.

Overview of the Control Program (CP)

CP acts as a hypervisor layer between the hardware and virtual machines. Each virtual machine appears to have its own CPU, storage (memory), and devices. In reality, these items can be

- Real. For example, you can dedicate a real network interface to a virtual machine for its exclusive use.
- Shared. For example, the CPU is shared through time sharing and real storage is shared as virtual storage. What appears as real storage to a guest is actually virtual storage to CP.
- Simulated. For example, a virtual switch is a simulated LAN networking switch.

CP transparently maps virtual devices and resources to their real counterparts. Topics in this section explain how CP manages computer resources for virtual machines.

Central processing units (CPUs)

A virtual machine can have up to 64 virtual CPUs. If capable of running in multiprocessor mode, your virtual machine operating system dispatches work on its virtual CPUs as if it were running on real hardware. CP handles the dispatching of work on your virtual CPUs to real CPUs.

Guideline: Never give a virtual machine more virtual CPUs than there are real CPUs.

Usually virtual machines share all CPUs, but a real CPU can be dedicated to a virtual machine, which means that the CPU is reserved for that virtual machine's exclusive use. This obviously has an impact on the performance of other virtual machines in the system.

Storage

Mainframe storage is analogous to memory in a personal computer. CP commands refer to memory as storage, so do not confuse the term "storage" with disk or tape storage.

Each virtual machine has its own virtual storage. CP manages the residency of virtual machine's pages in real storage through paging. Pages that have not been referenced can be moved out of real storage into either expanded storage or onto a paging device. When a virtual machine requires a page no longer in real storage, a page fault occurs and CP brings the missing page back into real storage.

CP has facilities that allow portions of real storage to be shared by many virtual machines. Such portions are called *shared segments*. This sharing economizes on real storage and requires less paging, thereby improving performance. For example, the CMS nucleus is shared in real storage by all virtual machines that loaded CMS by name; that is, every CMS virtual machine maps a 1 MB segment of virtual storage to the same 1 MB of real storage.

DASD and minidisks

DASD, the mainframe term for disk drives, stands for "direct access storage device" and is analogous to a hard disk drive on a personal computer. A single real DASD is called a *volume* or *real volume*. Each volume has a *label* or *volume serial number (volser)* that identifies the volume to z/VM.

Of special importance is the way z/VM shares DASD. CP can logically partition real DASD volumes into *minidisks*, which is analogous to dividing a personal computer hard disk into multiple partitions. A minidisk has its own label, which is distinct from the real DASD label. Each virtual machine can have one or more minidisks and those minidisks are under control of the guest operating system. To the guest, a minidisk appears as an entire DASD volume (though smaller) and the guest runs channel programs as normal to do I/O. Behind the scenes, CP reorients the channel programs: the guest perceives all minidisks as starting at cylinder 0, but the real DASD volume has only one cylinder 0, so CP must modify the cylinder offsets in the channel program to address DASD cylinders owned by the guest.

Temporary minidisks

You can create a temporary minidisk from a special pool of real disks. The disk lasts as long as the virtual machine is logged on. At logoff, the temporary minidisk is deleted and the space returned to the available temporary disk pool.

Virtual disks in storage

Virtual disks in storage are similar to temporary minidisks, except the disks are mapped to storage rather than the cylinders of real disks. Using virtual disks in storage avoids the need for disk I/O. CP manages the virtual disk pages as part of its real memory management.

Virtual readers, punches, and printers

These devices are not associated with real devices, but are implemented through the spool file system. For more information, see "Overview of the CP spool file system" on page 7.

The virtual machine console

The virtual machine console or *virtual console* is the primary interface to the virtual machine. When you log on to a virtual machine from a local terminal or a remote workstation, the virtual console is associated with the terminal session. From the

console, you can enter CP commands, such as loading (IPL) an operating system. The virtual console is the device an operating system views as its system or hardware console.

Note: The key assignments for your keyboard might differ from standard key assignments. Some 3270 emulators allow you to remap the key assignments; for example, the Clear function might be assigned to the ESC key on some keyboards and the Pause/Break key on others. Consult your display documentation for key mappings.

As you do work on your console, the lower right corner of the screen displays various status notices. The notices tell you what is happening in the system at the present time. If you forget what these notices mean, you can come back to this section for reference.

CP READ
This notice means that the Control Program (CP) is waiting for you to enter a command.

VM READ
This notice means that a virtual machine operating system, such as CMS, is waiting for you to enter a command.

RUNNING
This notice means the virtual machine is working on something. For CMS, this means CMS is ready for you to enter a command.

MORE ...
This notice means that there is more information than can fit on the current screen. After a pause (which depends on the terminal settings for your virtual machine), the next screen of information is displayed. To view the next screen right away, press the Clear key. To hold this information on the screen, press the Enter key, which changes **MORE...** to **HOLDING**.

HOLDING
This notice means the system is waiting for you to clear the screen before showing you more information. The notice appears when the screen displays **MORE...** and you press the Enter key. The notice can also appear when another user sends you a message. To cancel the hold, press the Clear key.

NOT ACCEPTED
This notice means that the system is working on something and is too busy to accept another command. Wait several seconds and issue your command again.

Related information

For more information about virtual consoles, see "Using a Virtual Machine Operator's Console" in *z/VM: Virtual Machine Operation*.

CP commands

CP provides you with commands that allow you to view and manipulate your virtual hardware (virtual CPUs, virtual storage, minidisks, and other devices). To issue some CP commands, you need to be in a special privilege class assigned to you in the user directory. Privilege classes are denoted by the letters A through Z, the numbers 1 through 6, or the word "Any." For the tasks explained in this document, the user IDs you use have all the required privilege classes (like superusers in Linux).

Related information

For more information about privilege classes, see "Privilege Classes" in *z/VM: CP Commands and Utilities Reference*.

Examples of using the CP commands: QUERY displays information about your virtual machine.

1. To display virtual CPUs, class G users can issue the QUERY VIRTUAL CPUS command:

```
query virtual cpus
CPU 00  ID  FF05152120640000 (BASE)
Ready;
```

   The response tells you the virtual machine has one base virtual CPU whose address is 00.

2. To display available storage (memory), class G users can issue the QUERY STORAGE command:

```
query virtual storage
STORAGE = 512M
Ready;
```

   The response tells you the virtual machine has 512 megabytes of storage.

3. To display information about minidisks, class G users can issue the QUERY VIRTUAL DASD command:

```
query virtual dasd
DASD 0120 3390 PK5001 R/O        250 CYL ON DASD  810B SUBCHANNEL = 000D
DASD 0190 3390 SYGEMC R/O        130 CYL ON DASD  7355 SUBCHANNEL = 0005
DASD 0191 3390 USGE24 R/W         10 CYL ON DASD  7378 SUBCHANNEL = 0000
DASD 019A 3390 PK5001 R/O        400 CYL ON DASD  810B SUBCHANNEL = 0009
DASD 019D 3390 US7E0K R/O        250 CYL ON DASD  801C SUBCHANNEL = 0006
```

   The response tells you the virtual machine has 5 minidisks, one of which is read/write (R/W); the others are read only (R/O).

4. To display information about network devices such as the Open Systems Adapter, class G users can issue the QUERY VIRTUAL OSA command:

```
query virtual osa
OSA  F5F0 ON OSA   F599 SUBCHANNEL = 0000
     F5F0 DEVTYPE OSA         CHPID 76 OSD
     F5F0 QDIO-ELIGIBLE       QIOASSIST-ELIGIBLE
OSA  F5F1 ON OSA   F59A SUBCHANNEL = 0001
     F5F1 DEVTYPE OSA         CHPID 76 OSD
     F5F1 QDIO-ELIGIBLE       QIOASSIST-ELIGIBLE
OSA  F5F2 ON OSA   F59B SUBCHANNEL = 0002
     F5F2 DEVTYPE OSA         CHPID 76 OSD
     F5F2 QDIO-ELIGIBLE       QIOASSIST-ELIGIBLE
```

5. To display the size of real storage, class B and E users can issue the QUERY STORAGE command:

```
query storage
STORAGE = 512M
```

Note: QUERY VIRTUAL *option* displays information about the virtual machine. The keyword "VIRTUAL" is optional for the class G user. For privileged

users (those with privilege classes other than G), using QUERY without the keyword "VIRTUAL" displays information about the real machine. For instance, QUERY VIRTUAL STORAGE displays the virtual storage size of the virtual machine while QUERY STORAGE (class B and E) displays the real machine storage size.

Related information

z/VM: CP Commands and Utilities Reference, SC24-6175

Overview of the CP spool file system

In the early days of computing, input to the computer came from punched cards loaded into a card reader. You used a key punch to record your program on punched cards, then loaded the cards into a card reader, which interpreted your cards and loaded your program into the computer. Output from the program was written to a printer. z/VM preserves this bit of computing history through virtual reader, punch, and printer devices, also called *unit record devices*. Unit record devices provide a handy way to send files from one virtual device to another, to other virtual machines, or to real devices (such as real printers). For instance, you can think of a file being sent from one virtual machine to another as the virtual equivalent of taking a card stack from one computer and loading the stack onto another computer's card reader.

Behind the manipulation of these files is a CP file system called the *spool file system*. CP manages spool files on one or more DASD volumes that act as temporary storage areas. A spool file is a collection of data along with device control instructions for processing on a unit record device. Spooling is the processing of files created by or intended for virtual readers, punches, and printers. Through CP and CMS commands, you can send spool files from one virtual device to another, from your virtual machine to another, and to real devices.

By convention, each virtual machine has a virtual reader at virtual device number 00C, a virtual punch at virtual device number 00D, and a virtual printer at virtual device number 00E. Your virtual reader is like the in-box of an e-mail system, except more than just e-mail can be placed there. Through your virtual punch, you can place a copy of an entire operating system into the system spool, then use the CP IPL command to load and run that operating system in your virtual machine. "Installing Linux in a virtual machine" on page 77 shows you how to use this z/VM facility.

Some important commands that operate on spool files are:

- SPOOL. Use the CP SPOOL command to set control options for one or more of your virtual spool devices. A handy way to keep a log of your system activity is to spool your console (SPOOL CONSOLE *, meaning send the console log to yourself), which keeps all your console activity in a spool file. When you close your console (SPOOL CONSOLE STOP CLOSE), your console log is sent to you.
- QUERY READER ALL. This CP command lets you view information about spool files in your virtual reader.
- RDRLIST. This CMS command displays information about your reader files in a full-screen interactive display.
- RECEIVE. This CMS command moves a file from your reader onto a minidisk.
- PUNCH. This CMS command punches (copies) a CMS file to your virtual punch.

Related information

For information about managing spool files for the entire z/VM system, start with the summary topics on controlling spool files in *z/VM: System Operation*, SC24-6233:

- Control Spool Files in the Print Queue
- Control Spool Files in the Reader Queue
- Control Spool Files in the Punch Queue

For information about managing spool files for your virtual machine, see "Using Spooled Devices to Print, Punch, and Read Information," in *z/VM: Virtual Machine Operation*, SC24-6241.

For command help, see *z/VM: CP Commands and Utilities Reference*, SC24-6175, and *z/VM: CMS Commands and Utilities Reference*, SC24-6166.

For online help, type **help** on the CMS command line, then press the Enter key.

The user directory

The z/VM *user directory* (or user registry) describes the configuration and operating characteristics of each virtual machine that can be created by CP. A z/VM user directory exists in two forms: a source form that consists of one or more CMS files, and an object form, compiled from the source, on a CP-formatted disk.

Each virtual machine has a *directory entry*. Here is a sample directory entry. The callouts in reverse type next to each statement correspond to explanations that follow the sample.

Note: In this document the user directory is modified by using the IBM Directory Maintenance program, DirMaint™, which handles both source and object forms of the user directory. Information about the directory entries is shown for educational purposes only. Unless explicitly instructed to do so, do not attempt to update the user directory source files manually.

```
1 USER LINUXC MYPASS 256M 1G G
2 IPL CMS
3 MACHINE ESA 4
4 CONSOLE 0009 3215
5 NICDEF BC0 TYPE QDIO LAN SYSTEM VSWITCH1
6 SPOOL 000C 3505 A
  SPOOL 000D 3525 A
  SPOOL 000E 1403 A
7 LINK MAINT 0190 0190 RR
  ...
8 MDISK 0191 3390 1595 50 VMLU1A MR
  MDISK 0200 3390 0001 3338 LX1519 MR
  MDISK 0201 3390 0001 3338 LX1559 MR
  MDISK 0202 3390 0001 3338 LX1599 MR
```

1. The USER statement begins a directory entry. The user ID for this virtual machine is LINUXC. "MYPASS" is the user's logon password. The virtual machine has a default storage of 256 megabytes ("256M"), but you can redefine storage up to a maximum of 1 gigabyte ("1G"). The second "G" means the virtual machine user is a general class user and can control functions for this virtual machine only.
2. The IPL statement indicates which operating system to load when you log on to the virtual machine. The example shows that CMS will be loaded. Loading CMS is handy because it allows you to make changes to the normal

environment as well as run some REXX™ EXECs (script-like executable files) to set up Linux. After changing the environment, you can load Linux into the virtual machine.

3. The MACHINE statement describes the processor architecture of the virtual machine. The maximum number of virtual CPUs that can be defined for this virtual machine is four. The default is one.
4. The CONSOLE statement defines the operating console (*virtual console*) for the virtual machine. CMS requires console type 3215. If supported by the operating system, you can specify 3270 or issue the CP command TERMINAL CONSOLE 3270 in the PROFILE EXEC prior to loading the operating system.
5. The NICDEF statement defines this virtual machine's attachment to a z/VM virtual switch.
6. SPOOL statements define the unit record devices. By convention, device number 000C is for the virtual reader (type 3505), device number 000D is for the virtual punch (type 3525), and device number 000E is for the virtual printer (type 1403).
7. LINK statements provide access to another virtual machine's minidisks.
8. MDISK statements define minidisks owned by the virtual machine. The format of the statement is:

   ```
   MDISK devno type start_cyl extent vol_label access_mode
   ```

   where

   devno
   Is the virtual device number of the minidisk.

   type
   Is the disk type of the real disk; typically 3390.

   start_cyl
   Is the real disk starting location of the first cylinder of the minidisk.

   extent
   Is the minidisk size in cylinders.

   vol_label
   Is the volume label of the real disk.

   access_mode
   Is the access mode. MR means the virtual machine has read/write access.

Related information

"Creating and Updating a User Directory," in *z/VM: CP Planning and Administration*, SC24-6178

Overview of the Conversational Monitor System (CMS)

Just as you can interact with Linux or UNIX® through a bash or Korn shell, you can interact with z/VM through CMS. Like a shell, you can use CMS to edit files, run EXECs (script-like executable files) or programs, modify the virtual machine environment, or modify z/VM itself. CMS is to z/VM as a shell is to Linux or UNIX.

Minidisks and the CMS access mode

CMS, like other operating systems running in a virtual machine, can access minidisks to store and retrieve files. For CMS, each minidisk has an *access mode*

represented by an alphabetic letter that determines how CMS searches for files. In Linux, path variables defining directories determine the search order for files. CMS searches for files among minidisks based on the alphabetical order of the access mode. First, CMS looks on the A minidisk, then the B minidisk, and so forth.

The 191 minidisk holds a special place in CMS. A 191 minidisk to a CMS user is like the home file directory for a Linux user. CMS always tries to access a user's 191 minidisk as access mode A. The CMS 191 minidisk is often called the "A-disk."

To see your CMS minidisks and their access modes, use the QUERY ACCESSED command. QUERY ACCESSED is similar to the `df` command in Linux. To access minidisks that are not already in the CMS access order, use the ACCESS command.

Example of viewing and accessing CMS minidisks

1. To view your accessed CMS minidisks, type the QUERY ACCESSED command and press the Enter key:

```
Ready;
query accessed
Mode  Stat     Files  Vdev  Label/Directory
A      R/W       595  191   CHA191
E      R/O      1776  201   IDTOOL
S      R/O       690  190   CMS21
Ready;
```

   The column under "Mode" shows the access mode for each minidisk. In the example, there are three minidisks accessed as A, E, and S.

   Notice that while in CMS all commands end with a "Ready;" prompt, indicating that CMS is ready to do more work.

2. To assign an access mode, use the ACCESS command. **Example:** To access the minidisk at virtual address 491 as B, type this command and press the Enter key:

```
Ready;
access 491 b
DMSACP723I B (491) R/O
Ready;
```

   The response tells you minidisk 491 is accessed read only (R/O) as B.

3. If you assign a mode currently assigned to another minidisk, the new minidisk replaces the current minidisk:

```
Ready;
access 19d d
DMSACC724I 19D replaces D (200)
DMSACP723I D (19D) R/O
Ready;
```

4. To remove a minidisk from an access mode, use the RELEASE command:

```
Ready;
release b
Ready;
```

CMS files

CMS files have a *file name*, *file type*, and *file mode*. File names and file types can be up to 8 characters long. The file mode corresponds to the access mode of the minidisk.

Examples:

```
PROFILE EXEC A1
MYDOC LISTING A1
DNFPFS LISTPS B1
```

By convention, some file types have special meanings. For example, EXEC is the file type for a file that contains executable statements, LISTING is the file type for text files, and LISTPS is the file type for PostScript® files.

To view and manipulate files, use the FILELIST command. FILELIST is similar to the `dir` command in Linux.

Examples of using FILELIST

1. To view all the files on your A-disk, type this command and press the Enter key:

   ```
   filelist
   ```

   Result: You see something like this:

   ```
    CHASTING FILELIST A0  V 169  Trunc=169 Size=253 Line=1 Col=1 Alt=0
   Cmd   Filename Filetype Fm Format Lrecl    Records     Blocks   Date     Time
         CHASTING NETLOG   A0 V         108       2132         53 10/15/03 16:02:30
         KIJL0CMD HGENRPT  A1 V         119         13          1 10/13/03 12:00:40
         KIJL0CMD LOG      A1 V         122        131          2 10/13/03 12:00:37
         KIJL0CMD SCRIPT   A1 V          81        454          4 10/13/03 12:00:37
         REXEC    HELPTCPI A1 V          79        133          2 10/13/03 10:26:11
         NETSTAT  HELPTCPI A1 V          79        749          9 10/13/03 10:25:31
   ```

2. In the “Cmd” column, you can type commands that are issued against the file on that line.

   Example: To edit a file in the filelist, type the XEDIT command in the “Cmd” column:

   ```
    CHASTING FILELIST A0  V 169  Trunc=169 Size=253 Line=1 Col=1 Alt=0
   Cmd   Filename Filetype Fm Format Lrecl    Records     Blocks   Date     Time
   xedit CHASTING NETLOG   A0 V         108       2132         53 10/15/03 16:02:30
         KIJL0CMD HGENRPT  A1 V         119         13          1 10/13/03 12:00:40
         KIJL0CMD LOG      A1 V         122        131          2 10/13/03 12:00:37
         KIJL0CMD SCRIPT   A1 V          81        454          4 10/13/03 12:00:37
         REXEC    HELPTCPI A1 V          79        133          2 10/13/03 10:26:11
         NETSTAT  HELPTCPI A1 V          79        749          9 10/13/03 10:25:31
   ```

3. Use “/” and “=” to avoid extra typing when you enter a command in FILELIST. The “/” means “this file” and “=” can be used to repeat a file name, file type, or file mode.

   Example: To copy a file called REXEC HELPTCPI from minidisk A to minidisk D, type this command and press the Enter key (typing over the other columns is OK):

```
 CHASTING FILELIST A0  V 169  Trunc=169 Size=253 Line=1 Col=1 Alt=0
Cmd   Filename Filetype Fm Format Lrecl    Records     Blocks   Date     Time
      CHASTING NETLOG   A0 V         108       2132         53 10/15/03 16:02:30
      KIJL0CMD HGENRPT  A1 V         119         13          1 10/13/03 12:00:40
      KIJL0CMD LOG      A1 V         122        131          2 10/13/03 12:00:37
      KIJL0CMD SCRIPT   A1 V          81        454          4 10/13/03 12:00:37
copy / = = d  HELPTCPI A1 V          79        133          2 10/13/03 10:26:11
      NETSTAT  HELPTCPI A1 V          79        749          9 10/13/03 10:25:31
```

4. To see only certain files, use "*" as a wildcard character.

   Example: To find any file on any accessed disk with a file type SCRIPT, type this command and press the Enter key:

```
filelist * script *
```

   Result: You see something like this:

```
 CHASTING FILELIST A0  V 169  Trunc=169 Size=555 Line=32 Col=1 Alt=0
Cmd   Filename Filetype Fm Format Lrecl    Records     Blocks   Date     Time
      APLANBD  SCRIPT   A1 V          65         20          1  7/16/02 12:31:01
      APROGBD  SCRIPT   A1 V          80        213          3  7/16/02 12:30:05
      B2HSYS   SCRIPT   Q1 V         113       4910         45  6/17/02 10:42:25
      B2HMSG   SCRIPT   Q1 V          76        670          7  6/17/02 10:42:04
      B2H      SCRIPT   Q1 V          72        107          1  5/20/02  0:47:02
      B2HAPP   SCRIPT   Q1 V          86       3129         25  5/20/02  0:47:02
      B2HEXA   SCRIPT   Q1 V          93       1390         10  5/20/02  0:47:02
      B2HINF   SCRIPT   Q1 V          81       1389         14  5/20/02  0:47:02
      B2HSETUP SCRIPT   Q1 V          70        175          2  5/20/02  0:47:02
      B2HUSE   SCRIPT   Q1 V          89       2622         25  5/20/02  0:47:02
      ACRONYMS SCRIPT   V1 V         962      62886        769  4/05/01 16:27:39
      VMSERVE  SCRIPT   V1 V         103       3180         31  1/24/01  8:48:49
```

The PROFILE EXEC

The PROFILE EXEC is a special executable file analogous to the .profile (or .bash_profile) in Linux and UNIX. Every time a CMS user logs on, CMS runs the PROFILE EXEC residing on the 191 minidisk, file mode A. You can use the PROFILE EXEC to set up your virtual machine environment; for instance, access disks, set up special PF keys, or even load another operating system in your virtual machine. In Chapter 7, "Creating your first Linux virtual machine and installing Linux," on page 71, you learn how to set up a PROFILE EXEC for your Linux virtual servers.

There can be times when you do not want the PROFILE EXEC to execute when you log on. For example, assume your PROFILE EXEC automatically loads Linux. If you have just shut down Linux and want to start CMS, but prevent Linux from being loaded again, you can prevent CMS from executing the PROFILE EXEC by issuing **access (noprof**. When you IPL (load) CMS, you see an identifier line displayed and CMS pauses with VM READ in the lower right corner of the display. At that point you can issue **access (noprof**:

```
IPL CMS
z/VM V6.1.0    2004-09-30 16:24









access (noprof
                                                        VM READ   GDLVME
```

The Help system

z/VM provides online help through the CMS Help system. The HELP command is like the man command in Linux. You can find full descriptions of z/VM commands by using the HELP command. By issuing **help**, you can access the main help menu for z/VM:

```
HELP TASKS                Task Help Information                 line 1 of 39
(c) Copyright IBM Corporation 1990, 2003

z/VM Help, main panel

This panel lists other Help panels that provide information about
various z/VM functions, topics, and tasks.
To view a Help panel, move the cursor to any character of the name
and press the ENTER key or the PF1 key.


HELPINFO - HELP Facility topics
MENUS    - z/VM Help menus
TASKS    - Basic z/VM tasks - good choice for beginners
COMMANDS - z/VM commands available to general users
CMS      - CMS commands
CP       - CP commands
QUERYSET - QUERY and SET commands and subcommands
TCPIP    - TCP/IP commands
PF1= Help     2= Top      3= Quit     4= Return     5= Clocate   6= ?
PF7= Backward 8= Forward  9= PFkeys  10=           11=          12= Cursor

====>
                                                        Macro-read 1 File
```

To get quicker access to command information, you can issue the HELP command with one of the keywords you see in the main menu. **Example:** For quick access to the information about the CP IPL command, issue:

```
help cp ipl
```

Examples of using the HELP command

1. To get help for all the CP commands, type this command and press the Enter key:

```
help cp menu
```

   Result: You see a screen like this:

```
 CP MENU                 Menu Help Information                   line 1 of 32
 (c) Copyright IBM Corporation 1990, 2003

 Help for CP commands

 To display a Help panel, move the cursor to any character
 of the name and press the ENTER key or the PF1 key.
 An asterisk (*) preceding the name indicates a MENU panel.
 A colon (:) preceding the name indicates a TASK panel.


 *CPQUERY  *XLINK    CPACcess  DISCARD   LOCate    RESTART   SYStem
 *CPSET    *XSPOOL   CPCAche   DISConn   LOCATEVM  RETAIN    TAg
 *CPUTIL   :DUMPS    CPHX      DISPlay   LOCK      REWind    TERMinal
 *DEFINE   :DYNIO    CPLISTfi  DRain     LOGoff    SAVESEG   TRace
 *DELETE   :HELP     CPRELeas  DUmp      Logon     SAVESYS   TRANsfer
 *DETACH   #CP       CPTRAP    DUPlex    Message   SCREen    TRSAVE
 *DISABLE  ACNT      CPTYPE    ECho      MODify    SEND      TRSOurce
 *DISPLAY  ACTivate  CPU       ENable    MONitor   SET       UNCOUPLE
 PF1= Help     2= Top      3= Quit     4= Return     5= Clocate   6= ?
 PF7= Backward 8= Forward  9= PFkeys  10=           11=          12= Cursor

====>
                                                          Macro-read 1 File
```

2. To get help for a specific command (for example, CP QUERY), type this command and press the Enter key:

```
help cp query
```

   Result: You see a screen like this:

```
 CP QUERY                All Help Information                    line 1 of 11
(c) Copyright IBM Corporation 1990, 2003

 QUERY

 Purpose

 You can display various information about your virtual machine by using the
 QUERY command operands.

 For information on the individual QUERY command operands, press PF11.

* * * End of File * * *









 PF1=          2= Top      3= Quit     4= Return     5= Clocate   6= ?
 PF7= Backward 8= Forward  9= PFkeys  10=          11= Related  12= Cursor

====>
                                                          Macro-read 1 File
```

Related information

- *z/VM: CMS Primer*, SC24-6172

- For more advanced information, see *z/VM: CMS User's Guide*, SC24-6173.
- For online help, type **help** on the CMS command line, then press the Enter key.

The CMS file editor XEDIT

CMS provides a file editor called XEDIT, which is a not only a full-screen editor, but a powerful programming tool. XEDIT has functions similar to `vi` in Linux. This topic introduces you to basic editing functions.

To enter an editing session, use the XEDIT command. **Example:** To create a new file called MY FILE A, type this command and press the Enter key:

```
xedit my file a
```

Without any modifications, an editing screen looks like this.

```
 MY       FILE     A1  F 80  Trunc=80 Size=0 Line=0 Col=1 Alt=0   1
DMSXIN571I Creating new file:  2




                                3


 4
===== * * * Top of File * * *  5
      |...+....1....+....2....+....3....+....4....+....5....+....6....+....7... 6
===== * * * End of File * * *

                                3


====> 7
                                                    8  X E D I T  1 File
```

Numbers in the figure explanations match the reverse type call-outs in the figure:

1. File identification line. The first line displays the file name, file type, file mode and other file characteristics. "F 80" means the length of a line is fixed at 80 characters. "Trunc=80" means any characters beyond the 80-character length are truncated. "Size=0" means there are no lines in this file. "Line=0" means the current line is 0 (more about the current line in point 5 on page 16). "Col=1" is the position of the column pointer (more about the column pointer in point 6 on page 16). "Alt=0" means the file has had no alterations.
2. Message line. XEDIT communicates with you by displaying messages on the second and third lines.
3. File area. This part of the screen is available to display the file. You can make changes to the file by moving the cursor under any line and typing over the characters, or by using special keys to insert or delete characters. You can make as many changes as you want on the displayed lines before pressing the Enter key. When you press the Enter key, the changes are made to the copy of the file that is kept in virtual storage. The SAVE or FILE subcommand permanently records those changes on the copy of the file that resides on disk.

Because a file can be too long to fit on one screen, various subcommands scroll the screen so you can move forward and backward in a file. Scrolling the screen is like turning the pages of a book.

4. Prefix area. The prefix area is the five left-most columns on the screen, and it displays five equal signs (=====). Each line in the file has a prefix area. You can perform various editing tasks such as deleting a line by entering short commands, called prefix subcommands, in the prefix area of a line.
5. Current line. The current line is the file line in the middle of the screen (above the scale). It is highlighted, appearing brighter than the other file lines.

   The current line is important because most subcommands perform their functions starting with the current line. Naturally, the line that is current changes during an editing session as you scroll the screen, move up and down, and so forth. When the current line changes, the line pointer (not visible on the screen) moves. Many XEDIT subcommands perform their functions starting with the current line and move the line pointer when they are finished.
6. Scale. The scale appears under the current line to help you edit. It is like the margin scale on a typewriter.

   The vertical bar (|) in column one on the scale is the *column pointer*. Various subcommands perform their functions within a line starting at the column pointer, which you can move to different positions on the scale by using XEDIT subcommands. The current column is the column under which the column pointer is positioned.
7. Command line. The large arrow (====>) at the bottom of the screen points to the command input area. One way you communicate with the editor is to enter XEDIT subcommands on this line. You can type subcommands in uppercase or lowercase or a combination of both, and many can be abbreviated. For example, BOTTOM, Bottom, and b are all valid ways to type the BOTTOM subcommand (which scrolls the file to the bottom).
8. Status area. The lower right corner displays the current status of your editing session, for example, edit mode or input mode, and the number of files you are editing. The status area in the figure shows you are editing one file.

Tip: If you want to explore XEDIT and its capabilities, type “help” at the XEDIT command line, which opens the XEDIT help menu.

Input mode

By issuing the subcommand INPUT at the command line (you can abbreviate the subcommand as "i"), you enter input mode.

```
 MY       FILE     A1  F 80  Trunc=80 Size=9 Line=0 Col=1 Alt=0
DMSXMD573I Input mode:









* * * Top of File * * *
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....
_ 1




                                2



====> * * * Input Zone * * * 3
                                                      Input-mode 1 File
```

XEDIT places the cursor **1** at the beginning of the input zone **2**. The input zone is an area in which you can place data. You can start typing at the cursor and, when you reach the end of a line, press the new line (Enter) key to return the cursor to the beginning of the next line. If you press the new line key on a line without data, XEDIT returns the file to editing mode.

XEDIT blocks the command line **3** because you cannot enter subcommands while in input mode.

Example of using input mode

1. Issue this command:

   ```
   xedit my file a
   ```

2. On the command line, type **input** and press the Enter key.
3. Type this phrase, then press the Enter key:

   ```
   CP is the z/VM hypervisor
   ```

4. Type this phrase, then press the Enter key:

   ```
   CMS is the interactive component of z/VM
   ```

5. Type this phrase, then press the Enter key:

   ```
   XEDIT is the CMS editor
   ```

6. Press the Enter key.

Result: Your XEDIT screen looks like this:

```
 MY       FILE     A1  F 80  Trunc=80 Size=3 Line=3 Col=1 Alt=3
DMSXMD587I XEDIT:




===== * * * Top of File * * *
===== CP IS THE Z/VM HYPERVISOR
===== CMS IS THE INTERACTIVE COMPONENT OF Z/VM
===== XEDIT IS THE CMS EDITOR
      |...+....1....+....2....+....3....+....4....+....5....+....6....+....7...
===== * * * End of File * * *









====>
                                                          X E D I T  1 File
```

Tip: XEDIT changed all lowercase letters to uppercase. To prevent this, issue the subcommand `set case mixed` before you add text.

Overview of changing files

The simplest way to change a file is to type over text on a line. However, there are times when you want to add or delete data in a file. Special keyboard keys and XEDIT subcommands help you do that:

- Insert key. When you press the insert key, the XEDIT cursor changes shape. By placing the cursor on a line, you can insert text between existing letters.
- Delete key. By placing the cursor on a line and pressing the delete key, the character to the right of the cursor is deleted and the line closes up.
- ADD and INPUT prefix commands. By moving the cursor to the prefix area, typing “a” and pressing the Enter key, you create a new line in the file. If you want to add five lines, type “a5” in the prefix area.

  The INPUT prefix command behaves similarly.
- DELETE prefix command. By moving the cursor to the prefix area, typing “d” and pressing the Enter key, you delete a line. If you want to delete five lines, type “d5” in the prefix area.
- LOCATE subcommand. You can find strings in the file by using the LOCATE subcommand. XEDIT scrolls the file to the line in which the string occurs. The invocation is `l /string` or simply `/string`. **Example:** To find an occurrence of the word “interactive” in a file, issue this command from the XEDIT command line:

  ```
  ====> l /interactive
  ```

  or simply:

  ```
  ====> /interactive
  ```

Example of changing files

Assume you are still editing the file in "Example of using input mode" on page 17.

1. From the XEDIT command line, type this command and press the Enter key:

```
====> top
```

2. To prevent XEDIT from turning lowercase letters to uppercase, type this command on the command line, then press the Enter key:

```
====> set case mixed ignore
```

3. Move the cursor to the first prefix area, type "a", then press the Enter key.
4. On the new line, type this phrase:

```
z/VM has many components
```

5. Type over the next line so the letters are in their proper case:

```
CP is the z/VM hypervisor
```

6. Create two blank lines between the first and second lines by typing "i2" in the second prefix area and pressing the Enter key.
7. Delete one of the blank lines by typing "d" in the prefix area, then pressing the Enter key.
8. From the XEDIT command line, locate the word "EDITOR":

```
====> /editor
```

Result: You should see a screen like this:

```
 MY       FILE     A1  F 80  Trunc=80 Size=5 Line=5 Col=1 Alt=0




===== * * * Top of File * * *
===== z/VM has many components
=====
===== CP is the z/VM hypervisor
===== CMS IS THE INTERACTIVE COMPONENT OF Z/VM
===== XEDIT IS THE CMS EDITOR
      |...+....1....+....2....+....3....+....4....+....5....+....6....+....7...
===== * * * End of File * * *









====>
                                                          X E D I T  1 File
```

SAVE, FILE, QUIT, and QQUIT

SAVE, FILE, QUIT, and QQUIT are XEDIT subcommands:

- Use SAVE when you want to save changes to a file permanently but continue editing the file.

- Use FILE when you want to save changes to a file permanently and quit editing the file.
- Use QUIT to quit editing a file you have not changed. If you have made any changes, XEDIT issues a message:

  ```
  DMSXSU577E File has been changed; type QQUIT to quit anyway
  ```

- Use QQUIT to quit a file and not save any changes you have made since the last save. The subcommand is handy if you decide you do not want any of the changes you have been currently making or if you want to be sure you have not changed a critical file.

Related information

- *z/VM: CMS Primer*, SC24-6172
- For more advanced information, see *z/VM: XEDIT User's Guide*, SC24-6245.

Summary of Linux and z/VM similarities

Though Linux and z/VM differ in many ways, they have similar functions and commands. Table 1 summarizes the similarities:

Table 1. Linux and z/VM similarities

Linux	z/VM (CP and CMS)
boot	IPL (initial program load)
`df` command	QUERY ACCESSED command
`dir` command	FILELIST command
file directory	disk access mode
kernel	Control Program (CP)
`man` command	HELP command
memory	storage
.profile	PROFILE EXEC
script	EXEC
shell	Conversational Monitor System (CMS)
user registry	user directory
`vi`	XEDIT

Chapter 2. Planning for Linux virtual servers

This topic covers system requirements and recommendations you need to follow **before** you install z/VM.

Figure 3 is an example of a mainframe configured for z/VM, Linux guests, and other operating systems.

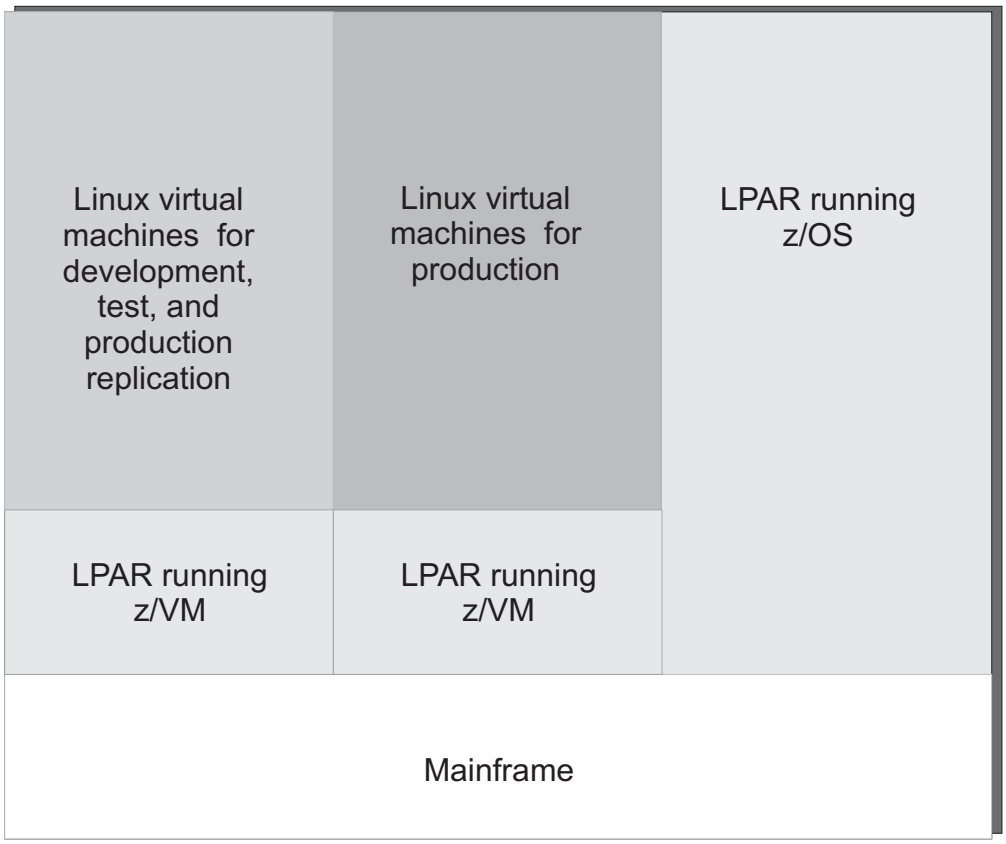

Figure 3. Example mainframe configuration

One logical partition (LPAR) is devoted to Linux production under z/VM. Another LPAR is devoted to Linux application development and test under z/VM; you might also run a replica of your production guests for testing purposes within this LPAR. Finally, another LPAR runs another operating system, such as z/OS.

The number of Linux guests you need to run depends on many factors. This topic discusses:

- Capacity requirements
- Memory and CPU requirements
- DASD space you need
- Network planning
- User management planning
- Obtaining your Linux distribution

Overview of z/VM capacity planning

An important element of z/VM capacity planning is knowing what z/VM is good at: the value of z/VM is its ability to consolidate distributed Linux workloads that under-utilize CPUs or do not require peak processing at the same time. z/VM improves the cost and performance efficiencies because it shares CPU cycles among virtual servers that, if distributed on separate servers, would be idle. There are three key characteristics that you should look for when deciding whether a Linux server could be consolidated on z/VM. Look for Linux workloads that:

- Under-utilize CPUs
- Do not require peak processing at the same time as others
- Have idle times, so that z/VM can share processing cycles with other Linux virtual servers.

Distributed servers running applications being considered for consolidation that run at high utilization throughout the day and peak with other candidate applications are poor candidates for consolidation. In general, the lower the utilization of a candidate application, or the more solitary its peaks are compared to other candidates under consideration, the more likely it can be consolidated.

Likewise, consider a benchmarking strategy that recognizes the real-world characteristics of your Linux workloads. A typical (inappropriate) approach is to conduct atomic measurements that compare throughput of a single instance of an application at a CPU utilization of 100%. This type of benchmarking practice, while simple and easy to conduct, yields inappropriate and misleading expectations of capacity because the practice does not incorporate any of the real-world operational characteristics or highlight any of the elements and advantages of consolidation. While such benchmark comparisons may be appropriate in a distributed paradigm for assessing capacity and performance of standalone servers running a single instance of an application, these comparisons are flawed when evaluating z/VM and the mainframe. The flaw is that such comparisons inflate the true operational utilization and throughput of the standalone distributed servers and do not account for the ability of z/VM to share idle cycles among virtual servers, which is not possible on under-utilized standalone distributed servers. Conducting a benchmark in such a fashion simply answers the question that, if you had one server running one instance of an application at an assumed utilization of 100%, how much throughput can you expect. In a consolidation case, that is not the question to ask. The question when designing a methodology for assessing capacity for consolidation is how many distributed workloads can you fit on z/VM using the true operational utilization and throughput of those workloads you are considering.

Once you have selected the right set of applications and their servers for consolidation, establish a base set of measurements that capture the real operational throughput of the servers. Figure 4 on page 23 shows a simplified consolidation example, in which there were many application instances running on separate standalone servers. Each of these application servers were 10% busy producing 74 transactions per second.

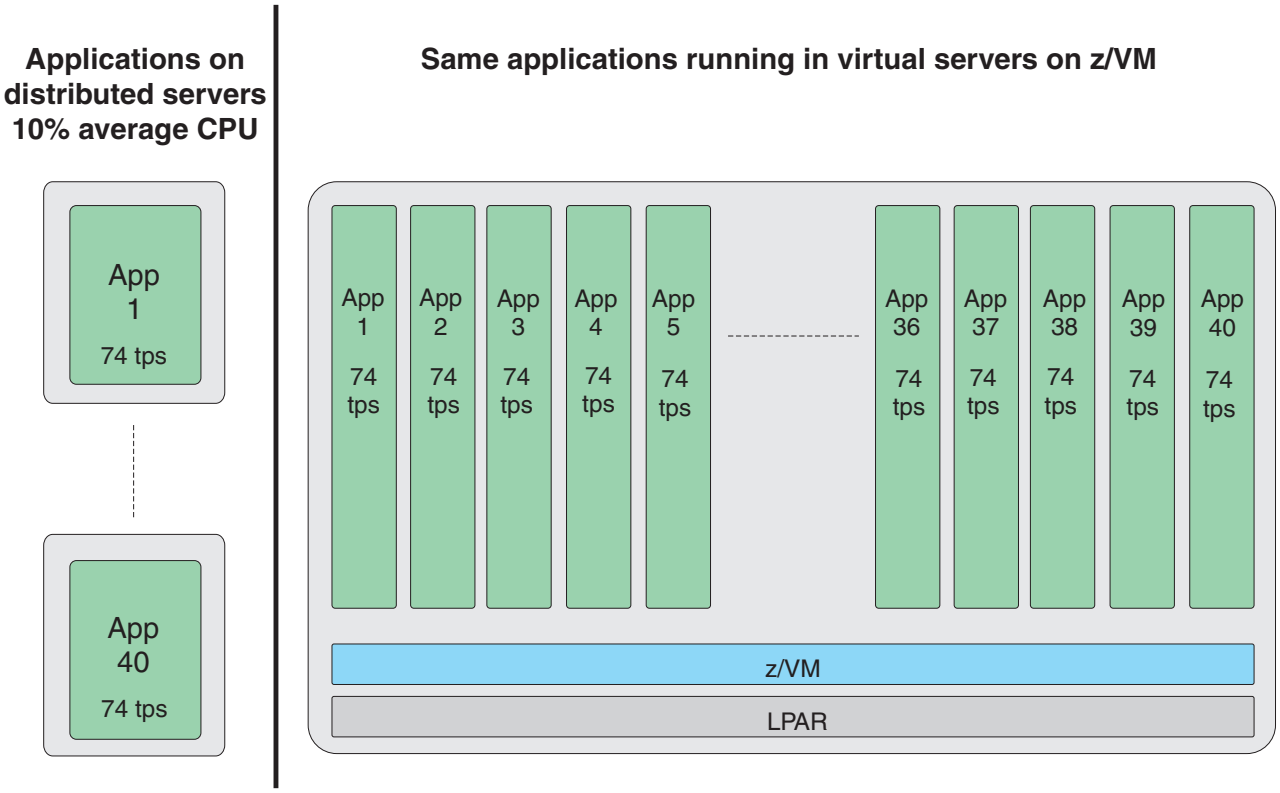

Figure 4. Server consolidation example

When you have established the baseline of 74 transactions per second for the distributed servers, define an equal number of z/VM virtual servers in which to run the applications.

To assess the system capacity required to support the same volume of work, tune the workload driver so that each instance of the application running in a virtual server produces the same transaction rate as its distributed counterpart.

The previous example showed an even distribution of work activity. However, the vast majority of real-world workloads skew the distribution of work. At any given moment, some applications are active while others are less active or idle. Unless your workload is evenly distributed, consider skewing the workload distribution as part of your capacity assessment.

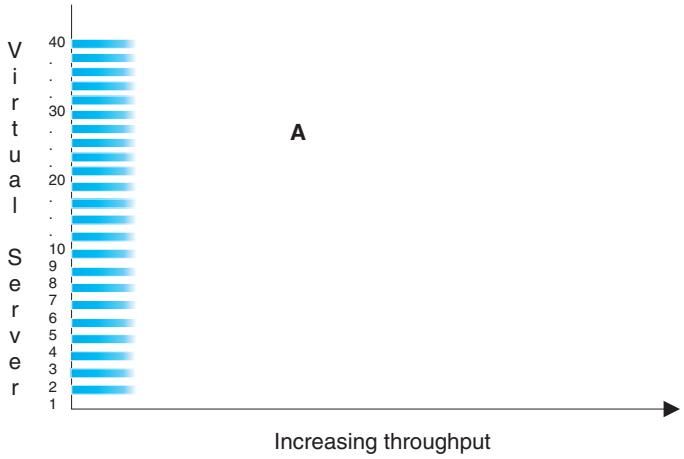

Figure 5. Workload distribution patterns (Part 1 of 3)

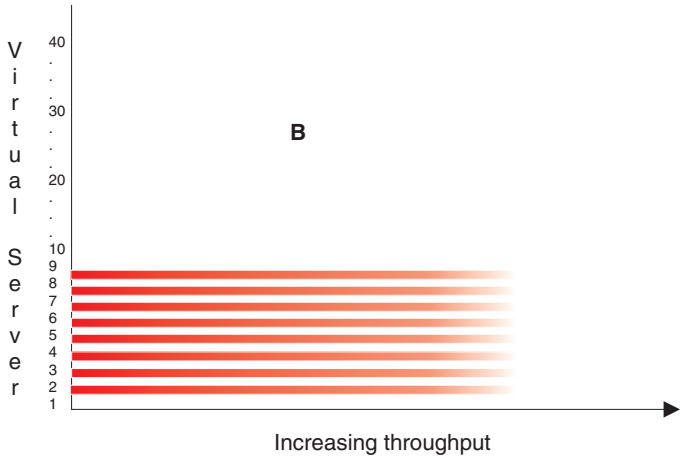

Figure 5. Workload distribution patterns (Part 2 of 3)

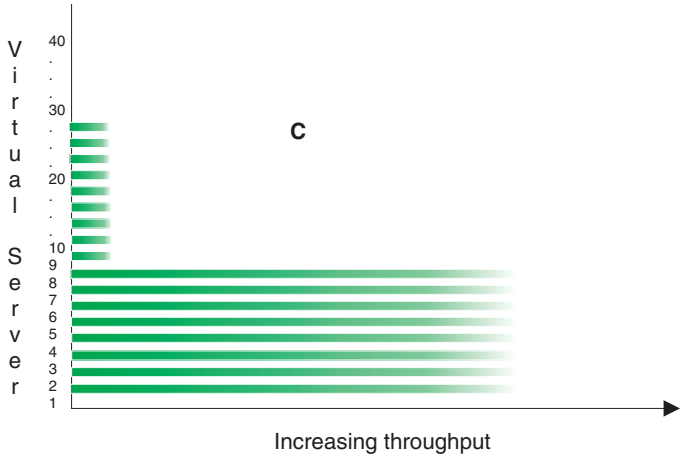

Figure 5. Workload distribution patterns (Part 3 of 3)

Figure 5 shows three workload distribution patterns. Workload distribution pattern A represents the prior example of an even distribution of work activity among the

applications. This pattern shows the worst-case, in which all workloads demand resources at the same time, rather than the characteristics of most production environments. Workload distribution patterns B and C show truer operational characteristics: at any given moment, some applications are busy while others are idle or less busy; and at different times, different applications are busy.

Figure 6 shows the relative throughput capacity of each of patterns A, B, and C.

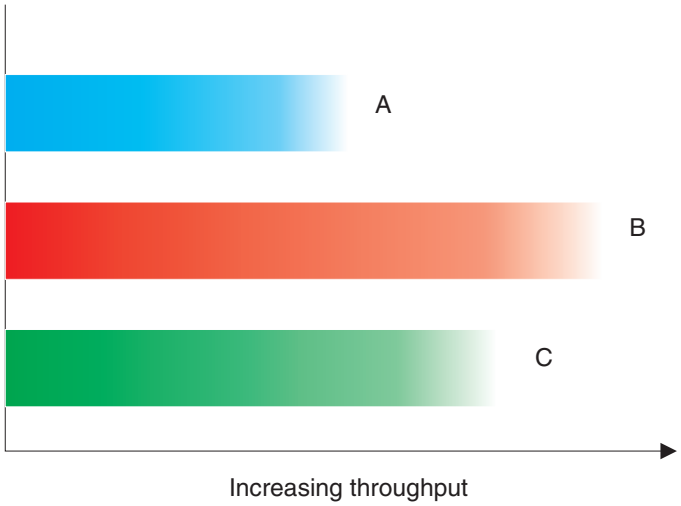

Figure 6. Relative throughput for patterns A, B, and C

Such distributions reflect the real world and place far less stress on the system because they are more cache-friendly and can result in sharply higher capacity results. Likewise, if your workload has characteristics of a skewed distribution, incorporate this aspect into your benchmarking methodology.

Estimating memory and CPU requirements

In most cases, initial system sizings are done with the assistance of IBM, your business partner, or consultant. This section gives you an appreciation for the things considered during an initial sizing and the things you should consider as you add work to your system.

To get you started, this topic gives you some basic knowledge about estimating the memory and CPU requirements for your Linux virtual servers. Such estimating is not an exact science and your experience may vary, but following the guidelines in this topic should help you get started, after which you need to measure the performance and fine tune your system. Topics in Chapter 11, "Monitoring performance and capacity," on page 117 help you fine-tune your initial configuration.

Overview of estimating memory and CPU requirements

Memory for the LPAR

A key factor in determining memory resources is the memory required for your applications. If the applications are new, you must estimate or start at some initial size; you can determine existing application memory requirements if the applications are currently running on other platforms. For example, you may not know how much memory a new WebSphere® application requires, so you can start

with 200 MB for the size of the Java™ Virtual Machine. Additionally, WebSphere Application Server requires 60 MB of memory.[1] So the total for your new application would be 260 MB. If you have an existing WebSphere application you know requires 250 MB of memory, the total with WebSphere Application Server would be 310 MB.

The total memory requirement for your applications, plus memory required for each Linux operating system and z/VM itself, give you an estimate of the memory required for a given LPAR. Do not pad the total memory figure.

Many customers prefer to isolate their applications by running each one in a separate virtual machine. Another strategy is to combine more than one application in a virtual machine, keeping the number of Linux guests to a minimum rather than creating one Linux guest for each application. The reason is that each Linux guest brings with it some overhead: each Linux operating system itself requires additional memory, and even an idle Linux guest uses some CPU resources. Also, applications sometimes can share middleware, which conserves memory. If it conforms to your installation's policy, you might be able to combine more than one application in a virtual machine, thereby saving memory. For example, several WebSphere applications can share the same WebSphere Application Server and the JVM in one virtual machine rather than each application having its own WebSphere Application Server and JVM in its own virtual machine, which would multiply the number of virtual machines and require more total memory.

Of the total memory requirement, start by allocating 75% to central storage and 25% to expanded storage. (Though z/OS no longer supports expanded storage, z/VM and the hardware do.) There are performance advantages to allocating expanded storage. Because z/VM is a 64-bit system, it seems as though allocating all memory to central storage makes sense. However, z/VM manages a paging hierarchy that uses expanded storage first, then slower DASD. Pages move to the slower paging DASD when not referenced within a given time limit. But, if pages are referenced again within the time limit, they can be brought back into central storage rapidly. This paging design gives a more consistent response to users.

During system operations, measure actual memory usage to test the initial memory allocation, which assumes all your guests need the estimated amount of memory all the time. Just as CPU demand has peaks and valleys, so does memory usage.

Memory for the virtual machines

For the general case of server consolidation, keep the virtual machine size small. How small you can define the virtual machine depends on the applications and workloads running in those virtual machines. Various Linux distributions might have minimum requirements as low as 64 MB. Some applications run fine in those minimum configurations. Other applications and workloads might require larger virtual machines. Avoid defining a virtual machine larger than it needs to be, because Linux uses excess memory for file and buffer caches. On a standalone system, these buffers can be very helpful for certain workloads to avoid I/O. However, in a virtual environment, the extra memory consumed adds to overall system memory contention. Such cases could cause a negative impact greater than the positive impact of I/O avoidance, which is especially true in configurations in which data is shared heavily between guests and is mostly read. In those configurations, z/VM minidisk caching can help avoid I/O. As a general guideline, define the virtual machine memory size to keep Linux on the verge of swapping.

1. Current requirements. For more information about Java and WebSphere memory requirements, consult Java and WebSphere documentation.

Lower the memory size until you see Linux begin to swap, then increase the virtual machine memory to the next bigger size.

Another way to reduce the memory requirements is through *discontiguous saved segments* (DCSS). A DCSS is an area of virtual storage outside the address range of a virtual machine. The area can contain read-only data or reentrant code. A DCSS connects discontiguous segments to a virtual machine's address space so programs can be fetched. DCSSs can be shared by many virtual machines, so total virtual memory required might be reduced. This reduced requirement for virtual memory can:

- Reduce CP paging requirements.
- Allow a given z/VM instance to support more Linux virtual machines.
- Reduce the amount of Linux disk I/O by having file systems, block devices, and shared objects in DCSSs.

A Linux guest machine uses DCSSs through the DCSS block device driver.

Related information

- This manual shows you how to set up a swap disk on a z/VM minidisk. Other options are available, such as using a virtual disk in storage. For more information on swap disk options, see "Linux Performance when running under VM" (http://www.vm.ibm.com/perf/tips/linuxper.html).
- For more information about DCSS block device drivers, see *Linux on System z: Device Drivers, Features, and Commands* on the IBM developerWorks® Linux on System z Web site entitled "Documentation for Development stream" at:

  http://www.ibm.com/developerworks/linux/linux390/documentation_dev.html

Real CPU requirements

The real CPU requirement is not simply a function of a single server: rather, the requirement is a function of all your virtual servers combined (see "Overview of z/VM capacity planning" on page 22). You must consider what each of your applications require and then estimate the overall CPU requirement. Your IBM representative, business partner, or consultant offer services to help you perform this task.

When defining z/VM LPARs, it is recommended that you assign a minimum of two logical processors. You have the option of dedicating the processors to the LPAR or sharing them with other LPARs. It is also possible to set different processing weights to LPARs, which gives more processor resources to one over another. For instance, you might give your production LPAR 60% weight and your test LPAR 40%.

Virtual CPU requirements

In general, follow this guideline: define as many virtual CPUs as needed (maximum CPU resources required), but do not exceed the number of real processors assigned to this LPAR. Extra virtual CPUs just add to the overhead and potentially increase the software multiprocessing factors.

Steps for estimating memory and CPU requirements

Before you begin: Read "Overview of estimating memory and CPU requirements" on page 25 to understand background topics.

Perform these steps to estimate memory and CPU requirements:

1. Determine a set of applications that will run in a Linux virtual server. The set might include one or more applications that process a given workload. For example, if you use a WebSphere application, you need to include the WebSphere application server and the JVM.
2. Determine the total memory requirement for the application set.
3. Add the memory requirement for the Linux operating system to the application set memory requirement. Enter the result in the left column in Table 2.
4. If you plan to run the same application set in more than one Linux virtual server (for instance, you want to have replicated virtual servers to distribute the workload), multiply the result from step 3 times the number of replicated servers. Enter the result in the right column in Table 2.
5. Repeat steps 1 through 4 for each application set.
6. Add up all the figures in the right column of Table 2, including the memory requirement for z/VM, to get a total.
7. Calculate the central and expanded storage sizes.
8. Record the number of real CPUs available, which helps you determine how many virtual CPUs each virtual machine has.

Table 2. Memory requirements worksheet

Application set plus Linux	x Number of virtual servers	x Memory estimate (MB)
1		
2		
3		
4		
5		
z/VM:		40
Total:		
Central storage estimate (Total x .75):		
Expanded storage estimate (Total x .25):		
Number of real CPUs:		

You are done estimating. Later, you will measure your system's performance and fine tune the configuration.

Guidelines for estimating the amount of DASD you need

Here are some guidelines on estimating the amount of DASD you need.

For z/VM paging

- Provide sufficient paging DASD for the paging subsystem. As shipped, z/VM has one full paging volume, which might be sufficient for your system. A rule of thumb is to provide twice as much DASD space as your total virtual storage

requirement. This value can be decreased by a fraction of the real memory available. See "Paging Space" in *z/VM: CP Planning and Administration*.

- Add paging space on a volume basis: do not use the paging volume for other purposes.

  "Steps for adding a paging, spooling, or user volume" on page 35 tells you how to add a paging volume.

Related information

For information about how z/VM uses DASD space and DASD space calculations, see "Direct Access Storage Requirements" in *z/VM: CP Planning and Administration*, SC24-6178.

For the Linux file system

- For disks for each Linux virtual server, the simplest guideline is to provide one 3390-3 at a minimum. A one-volume configuration supports many of the default packages installed by each Linux distribution. The need for additional DASD space depends on which additional products you install, the levels of those products, the size of user applications associated with those products, and end-user data.

  Linux views disk space in bytes while z/VM views disk space according to the device geometry (for example, number of cylinders). Regardless of the 3390 model, each cylinder holds approximately 670 KB when formatted. A 3390-3 has 3338 usable cylinders, which means it can hold about 2.2 GB when formatted. The difference between 3390 models is the number of cylinders the model has. A 3390-9 has three times the capacity of a 3390-3, so it holds approximately 6.6 GB.

  Consult product documentation for additional DASD space requirements. Your Linux distributor will tell you how much disk space to use for minimum and full installations.
- Eliminating certain Linux packages can help reduce your DASD requirements. Consider a minimal Linux installation to save disk space.
- Another strategy to save DASD is to divide up the Linux file system into read-only and read/write minidisks, then share the read-only minidisks among the virtual machines running Linux.
- Do not dedicate DASD to Linux (that is, do not allocate an entire DASD to Linux for its exclusive use). Instead, use minidisks for Linux volumes, especially those whose I/O operations are primarily read operations, to take advantage of minidisk caching. The z/VM Control Program by default maintains a minidisk cache for better performance. However, if there are many write operations to the minidisk, the extra Control Program overhead used to maintain the cache outweighs the benefits of caching. If the I/O operations include many write operations, turn off minidisk caching for the minidisk by using the CP command SET MDCACHE MDISK or the MINIOPT statement in the user directory.
- The Parallel Access Volume facility allows a controller to offer multiple device numbers that resolve to the same DASD, which allows I/O to the same DASD to happen concurrently. If you do not have the Parallel Access Volume facility, each DASD can do only one I/O at a time. To have your Linux file system perform well without the Parallel Access Volume facility, you can spread the file system across separate volumes and use the Linux logical volume manager, which maximizes the opportunity for the I/Os to the volumes to happen concurrently. For example, you could set up two stripes so Linux can do two I/Os at a time. Note that the volumes should be on separate controllers to avoid contention.

Planning your network

To use Linux, you need to connect to your TCP/IP network.

You need to determine:

- The device addresses of your network interface.
- The host name and domain name of your Linux virtual servers
- The IP address and the subnet mask
- Depending on the connectivity type, the broadcast name server and network addresses.
- Whether you plan to have your Linux virtual servers use the cryptographic facility for SSL acceleration.

TCP/IP networking options for Linux

z/VM provides two broad categories for TCP/IP connectivity for Linux guests:

- Real network interfaces with connections to the LAN. A real network connection may be through any device supported by Linux, including IBM Open Systems Adapters and channel-attached devices. The real device (as defined in the Input/Output Configuration Data Set) must be dedicated to the virtual machine running Linux. You can do this by providing DEDICATE entries in the CP directory entry for the virtual machine or by using the CP ATTACH command.
- Virtual network interfaces, which allow the real connections to be shared, maximizing throughput. Virtual network connections include:
  - Guest LAN. Through a guest LAN, z/VM simulates OSA-Express or HiperSockets™ microcode to allow you to connect guest systems to communication adapters. Such connections enable guests to communicate through a LAN rather than through point-to-point connections. If the guests require external connectivity, that connection requires a virtual machine acting as a router between the guest LAN and the external connection.
  - Virtual switch. A virtual switch is a special kind of guest LAN. In addition to providing a network of virtual adapters, the switch can be connected directly to an OSA-Express QDIO adapter. This capability allows you to gain connectivity to external LAN segments without requiring a router, reducing CPU utilization and latency associated with providing external connectivity through a router.

The virtual switch is the preferred way to connect your Linux machines to the network, but there are other legacy connection types. For more information, see "Networking Options in z/VM" in *z/VM: Connectivity*, SC24-6174.

Figure 7 on page 31 is a diagram of a virtual switch called VSWITCH1. Coupled to the virtual switch through NICDEF directory statements are Linux virtual servers. The DTCVSW1 service virtual machine[2], running a subset of the TCP/IP stack, is the virtual switch controller. The full TCP/IP stack runs in the TCPIP service virtual machine, the z/VM production TCP/IP. The two service virtual machines are kept separate so you can operate them independently.

Due to the advantages of virtual switches, this document shows you how to set up a virtual switch configuration only. IBM created the TCP/IP and user directory

2. A *service virtual machine* or *service machine* is virtual machine that provides a system service, such as accounting, error recording, or monitoring.

changes for the default virtual switch controllers DTCVSW1 and DTCVSW2[3], but there are other tasks you must do. These tasks are intermingled with other configuration tasks in Chapter 3, "Changing the system configuration," Chapter 5, "Configuring TCP/IP," and Chapter 7, "Creating your first Linux virtual machine and installing Linux." To place the tasks in context, Table 3 summarizes how to configure TCP/IP and a virtual switch.

Table 3. Task roadmap for configuring TCP/IP and a virtual switch

Subtask	Associated instructions (see . . .)
Define a virtual switch for z/VM	"Steps for defining a virtual switch" on page 44
Configuring the production TCP/IP	"Setting up the production TCP/IP" on page 63
Configuring TCP/IP to be logged on automatically	"Steps for automatically starting TCP/IP" on page 63
Connecting your virtual machine to the virtual switch	"Overview of defining virtual machines for Linux" on page 71
Configuring Linux to use the virtual switch	Your Linux installation documentation. For Linux, the connection is defined as an OSA-Express device.

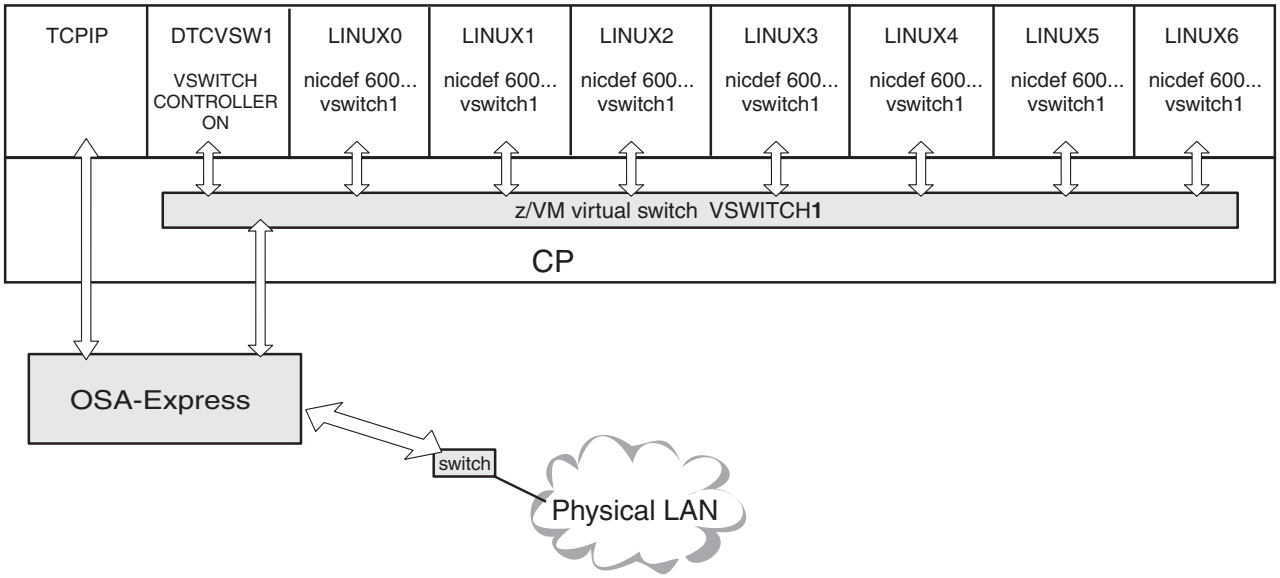

Figure 7. Example of a virtual switch

Related information

"Networking Options in z/VM" in *z/VM: Connectivity*, SC24-6174

Giving Linux virtual servers access to cryptographic hardware for SSL acceleration

To run Linux as a guest under z/VM with access to cryptographic hardware for SSL acceleration, you must

3. A second virtual switch controller is available for failover: CP automatically chooses another virtual switch controller should the first one fail.

1. Define the cryptographic facility for the LPAR in which z/VM runs through the Hardware Configuration Definition.
2. Define the cryptographic capability for each Linux virtual machine in the user directory.
3. Have the z90crypt device driver integrated into the Linux operating system. Some distributions have the device driver integrated, while other distributions require you to install it.

The user directory statement `CRYPTO APVIRT` provides access to the cryptographic hardware and allows the z90crypt device driver to use cryptographic instructions. z/VM manages a pool of hardware cryptographic queues that are shared among all the guests using the cryptographic facility. You can create more guests that share the cryptographic facility than the actual number of hardware queues available. Even though the hardware queues are shared, the data remains isolated and is not vulnerable or exposed to other Linux images.

"Steps for defining a master virtual machine for Linux" on page 71 shows you how to add the `CRYPTO APVIRT` user directory statement to the master Linux virtual machine, which means all replicas of this master have access to the cryptographic facility. If you prefer, you can leave this statement out of the master Linux virtual machine and add the user directory statement to individual Linux virtual machines only.

z/VM provides CP commands to manage the cryptographic facility. See "Step for managing real devices" on page 100, and "Virtual machine operation tasks" on page 107.

Related information

- For more information about defining the cryptographic facility for the LPAR in which z/VM runs, consult your hardware and Hardware Configuration Definition documentation.
- For more information about z/VM's support for the cryptographic facility, see "Using a Cryptographic Coprocessor Facility" in *z/VM: CP Planning and Administration*.
- For information about setting up secure SSL communications, see "Configuring the SSL Server" in *z/VM: TCP/IP Planning and Customization*.
- For information about the z90crypt device driver, see *Linux on System z: Device Drivers, Features, and Commands* on the IBM developerWorks Linux on System z Web site entitled "Documentation for Development stream" at:

  http://www.ibm.com/developerworks/linux/linux390/documentation_dev.html

Planning for user management

To add a new user to z/VM, you must create a directory entry for a new virtual machine. Through native facilities, you can update a file called USER DIRECT, then run the DIRECTXA utility to compile the source file and place the new user directory online. The USER DIRECT file is simply a CMS file containing various directory statements. A virtual machine definition is a grouping of directory statements beginning with a USER statement and ending with either the next USER statement or the end of the file.

You can administer the user directory by editing the USER DIRECT file, then placing the user directory online through the DIRECTXA command. However, such a method of user management is cumbersome and error prone. Because the user

directory is so important for z/VM, a corrupt or invalid online user directory can be disastrous. For example, if you inadvertently overlap minidisk definitions, it could cause serious data loss. If your z/VM system has more than a handful of virtual machines, it makes little sense to manage users manually. You need an automated facility.

The Directory Maintenance Facility for z/VM (often called DirMaint) is just such an automated facility. DirMaint is a CMS application that helps you manage your z/VM user directory through a simplified command interface and automated facilities. DirMaint commands, which are like their corresponding directory statements, initiate user directory transactions. DirMaint error checking ensures that only valid changes are made to the user directory, and that only authorized personnel are able to make the requested changes. Any transaction requiring the allocation or deallocation of minidisk extents can be handled automatically. You can control all user-initiated transactions through passwords and record transactions for auditing purposes.

When you activate DirMaint, you give control over the user directory to the DIRMAINT service virtual machine. The source USER DIRECT file on the MAINT virtual machine's 2CC disk is no longer valid and you must not use the DIRECTXA command. DirMaint maintains and updates the online user directory. You interact with the DIRMAINT service machine through commands to make changes to the user directory.

A handy DirMaint function for virtual machines is the capability to define template (or prototype) directory entries. With this function, you can clone a virtual machine with a few commands by instructing DirMaint to create a new virtual machine like the prototype entry and to replicate the prototype's disks. For more information about using templates for cloning virtual machines, see Chapter 8, "Cloning Linux virtual servers," on page 83.

DirMaint also has a service virtual machine called DATAMOVE. The service machine moves the contents of CMS-formatted minidisks from one disk to another, then erases the contents of the CMS minidisks being deleted. Because these functions are time consuming, the functions are offloaded from the DIRMAINT service machine to the DATAMOVE service machine, making DIRMAINT available to process commands.

Directory Maintenance Facility is preinstalled on your system in a disabled state. To use DirMaint, you must first pay a license fee, then enable the feature and configure it. Table 4 gives you an overview of the tasks involved for enabling and configuring DirMaint.

Table 4. Task roadmap for enabling and configuring DirMaint

Subtask	Associated instructions (see . . .)
Enabling the Directory Maintenance Facility	"Steps for enabling DirMaint" on page 51

Table 4. Task roadmap for enabling and configuring DirMaint (continued)

Subtask	Associated instructions (see . . .)
Configuring the Directory Maintenance Facility	• "Steps for changing the passwords for DirMaint service machines" on page 52 • "Steps for configuring DirMaint" on page 53 • "Steps for authorizing users to perform DirMaint tasks" on page 54 • "Steps for controlling where DirMaint creates minidisks" on page 55 • "Steps for copying the current USER DIRECT file" on page 57 • "Steps for putting the configuration into production and starting DirMaint" on page 58 • "Steps for automatically starting DIRMAINT" on page 59
Testing and using the Directory Maintenance Facility	• "Steps for testing DirMaint" on page 60 • "Step for modifying the OPERATOR's directory entry" on page 60

Related information

- *Program Directory for IBM z/VM Directory Maintenance Facility Feature* at: http://www.ibm.com/eserver/zseries/zvm/library/
- *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*, SC24-6190

Steps for obtaining documentation and media

You must obtain your Linux distribution documentation and installation media.

Before you begin: You need to decide which Linux distribution you will use.

Perform these steps to obtain documentation and media:

1. Obtain Linux documentation according to your distribution:
   - SUSE Linux documentation at http://www.suse.com
   - Red Hat, Inc., documentation at http://www.redhat.com
2. Access the installation media provided with your distribution. You need access to:
   - The Linux system files (kernel, parm file, and initial RAM file)
   - A UNIX-based FTP or NFS server attached to your network.

You are finished when you have obtained the documentation and access to installation media.

Chapter 3. Changing the system configuration

This topic tells you how to change the z/VM system by updating the SYSTEM CONFIG file.

Overview of the SYSTEM CONFIG file

The *SYSTEM CONFIG* file contains the primary system definitions used when CP is booted (IPLed). All of the information needed to configure CP statically comes from this file. As a system programmer, you should become familiar with this file.

The SYSTEM CONFIG file resides in the same location as the bootable CP kernel. Both objects, along with other files and modules, reside on a special CMS-formatted minidisk. Minidisks containing such objects are called parm disks because when allocated those disks are given a special record category type called "PARM". There can be more than one parm disk allocated in a z/VM system for backup and recovery. However, for any specific IPL, only one parm disk is used to load code and configuration files.

Related information

"Using Configuration Files" in *z/VM: CP Planning and Administration*, SC24-6178.

Steps for adding a paging, spooling, or user volume

z/VM classifies DASD volumes according to their use. In this topic, you format and allocate paging, spooling, and user volumes. A *paging volume* is a volume owned by CP that is used by the paging subsystem; a *spooling volume* is a volume owned by CP that is used by the spooling subsystem for spool files; and a *user volume* is a DASD volume that CP uses for minidisks.

Before you can attach a paging, spooling, or user volume, you must format and label the volume. Later, you will update the CP-owned volume list in SYSTEM CONFIG to include new paging and spooling volumes, and the user volume list in SYSTEM CONFIG to include user volumes, so you need to format the new volumes now. Use the CPFMTXA utility program. Then use CPFMTXA to allocate the volumes as paging, spooling, or user volumes.

Before you begin: You need to know how many paging, spooling, and user volumes you need. See "Guidelines for estimating the amount of DASD you need" on page 28.

You need to log on as MAINT. You need to have one or more real DASD volumes connected to your computer.

Perform these steps to format a paging, spooling, or user volume:

1. Attach the volume to MAINT.

   Example: If the volume is at real address 202, type this command and press the Enter key:

```
attach 202 *
DASD 0202 ATTACHED TO MAINT 0202 WITH DEVCTL
Ready;
```

2. Start CPFMTXA:

```
cpfmtxa 202
```

3. Type in these responses and press the Enter key in response to each prompt:

```
ENTER FORMAT, ALLOCATE, LABEL, OR QUIT:
format
ENTER THE CYLINDER RANGE TO BE FORMATTED ON DISK 0202 OR QUIT:
000 end
ENTER THE VOLUME LABEL FOR DISK 0202:
label
FORMAT WILL ERASE CYLINDERS 00000-00544 ON DISK 0202
DO YOU WANT TO CONTINUE? (YES|NO)
yes
.
.
.
ENTER INPUT COMMAND:
end
ICK00002I ICKDSF PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
ENTER ALLOCATION DATA
.
.
.
```

where *label* is a volume label (for instance, PAG002 for a paging volume, SPL002 for a spooling volume, or USR002 for a user volume).

4. In response to the "ENTER ALLOCATION DATA" prompt, type one of these responses and press the Enter key:
   - If you are creating a paging volume:

     ```
     page 0 end
     end
     ```

   - If you are creating a spooling volume:

     ```
     spol 0 end
     end
     ```

   - If you are creating a user volume:

     ```
     perm 0 end
     end
     ```

   Result: You see more messages from the program. Finally, you see this:

   ```
   ICK00002I ICKDSF PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
   ```

5. Detach the volume.

```
detach 202
DASD 0202 DETACHED
Ready;
```

6. Repeat these steps for other paging, spooling, and user volumes you need.

7. Record the volume label and allocation type for each volume you have defined. You will use this information later when updating the CP-owned volume list.

You are done for now. Later, you will make these volumes available to CP for its use.

Steps for releasing the primary parm disk

Before you begin updating the SYSTEM CONFIG file, you must end CP's access to the primary parm disk.

Before you begin: You need to log on as MAINT.

Perform these steps to release the primary parm disk:

1. Instruct CP to end its access to the primary parm disk. Type this command and press the Enter key:

```
cprelease a
CPRELEASE request for disk A scheduled.
HCPZAC6730I CPRELEASE request for disk A completed.
Ready;
```

2. Access the primary parm disk (MAINT's CF1). Type these commands and press the Enter key after each command:

```
link * cf1 cf1 mr
Ready;
access cf1 z
Ready;
```

You know you are done when you have access to the CF1 disk.

Steps for updating the CP-owned volume list

The *CP-owned volume list* is the place where you specify the labels of paging and spooling volumes that CP should automatically attach to the system during IPL. These volumes contain the system data for your z/VM system. All other volumes on the system are considered user volumes.

Before you begin: You need to format and allocate your paging and spooling volumes. See "Steps for adding a paging, spooling, or user volume" on page 35.

You need to end CP's access to the primary parm disk. See "Steps for releasing the primary parm disk."

Perform these steps to update the CP-owned volume list:

1. Edit the SYSTEM CONFIG file. Type this command and press the Enter key:

```
xedit system config z
```

Result: You see a file like this:

```
 SYSTEM   CONFIG   Z1  F 80  Trunc=80 Size=277 Line=4 Col=1 Alt=0

* * * Top of File * * *
/*********************************************************************/
/*             SAMPLE SYSTEM CONFIG FILE                             */
/*********************************************************************/
/*                                                                   */
/*  Rules of the config file:                                        */
/*                                                                   */
/*    1) REXX style comments are permitted                           */
/*                                                                   */
/*    2) Configuration commands can be continued to next line via a  */
/*       trailing comma on the previous line                         */
/*                                                                   */
/*    3) IMBED statements can be used to imbed other files that      */
/*       reside on the PARMDISK into the configuration file          */
/*                                                                   */
/*    4) The IMBED record is of the format:    IMBED fn ft           */
/*                                                                   */
/*    5) Tolerance record can be used to signal whether CP should    */
/*       tolerate errors in some sections of the CONFIG file or not; */
/*       default is to have tolerance on (that is to tolerate errors) */
/*                                                                   */
====>
```

2. Find the section titled "CP_Owned Volume Statements." At the XEDIT command line, type this command and press the Enter key:

```
====> /cp_owned volume statements
```

3. Move the cursor to a CP-owned slot marked "Reserved."
4. Type over the word "Reserved" with the volume label of a paging or spooling volume that you allocated in "Steps for adding a paging, spooling, or user volume" on page 35.

   Example: If you are adding a spooling volume called "SPL001," type in that volume label and use the space bar to remove the rest of the characters in the word "Reserved."
5. Move the cursor to the next line and repeat step 4 for the other paging and spooling volumes you have allocated.
6. Save the file. At the XEDIT command line, type this command and press the Enter key:

```
====> save
```

You should now see something like this:

```
 SYSTEM   CONFIG   Z1  F 80  Trunc=80 Size=277 Line=74 Col=1 Alt=0

/*********************************************************************/
/*                    CP_Owned Volume Statements                     */
/*********************************************************************/

    CP_Owned   Slot   1  610RES
    CP_Owned   Slot   2  610SPL
    CP_Owned   Slot   3  610PAG
    CP_Owned   Slot   4  610W01
    CP_Owned   Slot   5  610W02
    CP_Owned   Slot   6  SPL001
    CP_Owned   Slot   7  RESERVED
    CP_Owned   Slot   8  RESERVED
    CP_Owned   Slot   9  RESERVED
    CP_Owned   Slot  10  RESERVED
    CP_Owned   Slot  11  RESERVED
    CP_Owned   Slot  12  RESERVED
    CP_Owned   Slot  13  RESERVED
```

Steps for updating the default system identifier

The default system identifier appears on printed output separator pages and the status area of 3270 display screens. z/VM assigns the identifier ZVMV6R10 to all z/VM version 6 release 1 systems. If you have many z/VM systems, it is difficult to determine which z/VM system you are logged onto unless you make the system identifier unique for each system.

Before you begin: You need to end CP's access to the primary parm disk. See "Steps for releasing the primary parm disk" on page 37.

Perform these steps to update the default system identifier:

1. Edit the SYSTEM CONFIG Z file. Type this command and press the Enter key:

   ```
   xedit system config z
   ```

2. Find the line "System_Identifier_Default." At the XEDIT command line, type this command and press the Enter key:

   ```
   ====> /system_identifier_default
   ```

3. Replace the default system identifier with your own:

   ```
   System_Identifier_Default mysystem
   ```

   where *mysystem* is your system identifier, such as VM610A.

4. Save the file. At the XEDIT command line, type this command and press the Enter key:

   ```
   ====> save
   ```

You should now see something like this:

```
 SYSTEM   CONFIG   Z1  F 80  Trunc=80 Size=277 Line=114 Col=1 Alt=0

/*********************************************************************/
/*                  System_Identifier Information                    */
/*********************************************************************/

    System_Identifier_Default VM610A
```

Steps for updating the user volume list

Just as there is a list of DASD volumes that CP should automatically attach to the system during IPL for access to CP system areas, there is a list of DASD volumes that CP should automatically attach to the system for user minidisk definitions. Because all minidisks are managed by CP, all volumes that house minidisks must be attached to the z/VM system. CP must control the volumes so it can reorient channel programs initiated by a guest operating system. The guest perceives its disks as starting at cylinder 0, but the true location of the guest's minidisk starts at an offset of real cylinder 0.

If no user volumes are attached to the system at IPL time, the real devices housing minidisks need to be attached manually (see "Step for managing real devices" on page 100). Otherwise, virtual machines will have no disks. To avoid manual attachment, you can tell CP to look for DASD volume labels and attach those devices at IPL time.

The USER_VOLUME_LIST statement directs CP to attach specific user DASD volumes at z/VM load (IPL) time. The USER_VOLUME_INCLUDE statement allows you to create a general volume identifier and to include all volumes that match the general identifier. For example, if all your Linux user volumes have a volume identifier starting with V2LX, you can add this statement:

```
User_Volume_Include V2LX*
```

Tip: If a volume is normally attached to the system using a USER_VOLUME_INCLUDE statement, CP does not notify the operator if the volume is not mounted. If a user volume is necessary for normal system operation, specify it with a USER_VOLUME_LIST statement so that the operator is notified during system initialization if the volume is not mounted.

Before you begin: You need to format and allocate the user volumes you need. See "Steps for adding a paging, spooling, or user volume" on page 35.

You need to end CP's access to the primary parm disk. See "Steps for releasing the primary parm disk" on page 37.

Perform these steps to update the user volume list:

1. Edit the SYSTEM CONFIG Z file. Type this command and press the Enter key:

   ```
   xedit system config z
   ```

2. Find the section titled "User_Volume_List." At the XEDIT command line, type this command and press the Enter key:

   ```
   ====> /user_volume_list
   ```

3. Move the cursor to the “User_Volume_List” statements.
4. Blank out the comments (“/*” and “*/”) on either side of the “User_Volume_List” statement and add the DASD label for the user volume.

   Example:

```
User_Volume_List LINUSR
```

5. Move the cursor after the User_Volume_List statements. In the prefix area, type “i” and press the Enter key:

```
00107     User_Volume_List LINUSR
00108 /*  User_Volume_List USRP02    */
00109 /*  User_Volume_List USRP03    */
00110 /*  User_Volume_List USRP04 USRP05 USRP06 USRP07   */
i
```

6. Add a User_Volume_Include statement.

   Example:

```
User_Volume_Include V2LX*
```

7. If necessary, repeat steps 4, 5, and 6.
8. When you finish the User_Volume_Include statements, press the Enter key.
9. Save the file. At the XEDIT command line, type this command and press the Enter key:

```
====> save
```

You should now see something like this:

```
 SYSTEM   CONFIG   Z1  F 80  Trunc=80 Size=253 Line=101 Col=1 Alt=3

/*********************************************************************/
/*                          User_Volume_List                         */
/* These statements are not active at the present time.  They are    */
/* examples, and can be activated by removing the comment delimeters. */
/* Multiple labels can be stacked on one statement.                  */
/*********************************************************************/

    User_Volume_List LINUSR
/*  User_Volume_List USRP02    */
/*  User_Volume_List USRP03    */
/*  User_Volume_List USRP04 USRP05 USRP06 USRP07   */

    User_Volume_Include V2LX*
```

Steps for setting up warm start, clearing tdisk space, and other features

The FEATURES statement in SYSTEM CONFIG allows you to modify attributes associated with the running system at IPL time. In this procedure, you will modify some of the features:

- The Auto_Warm_IPL feature causes CP to bypass prompting for start options, provided the previous system shutdown was successful. The feature allows for a fully automated startup of z/VM.

- The Clear_TDisk feature causes CP to erase temporary disks fully (that is, overwrite the entire temporary disk with zeros) after those disks are detached. The feature prevents another user who may define an identically sized temporary disk from accessing data written by the previous user.
- The Retrieve defines the default and maximum number of retrieve buffers allowed per user on your system. Retrieve buffers create a command history, from which users can retrieve commands previously issued. Command retrieval is usually assigned to a program function key such as PF12 (F12). The assignment is through the CP SET command, SET PF12 RETRIEVE. By pressing PF12, a command is retrieved and written back into the command area on the terminal screen. You probably do not need to change these settings.
- The Passwords_on_Cmds feature tells CP whether to prompt users for passwords when using the CP AUTOLOG, LINK, or LOGON commands.
- The Disconnect_timeout feature controls whether and when a virtual machine is logged off after it has been forced to disconnect. You will turn this feature off, so that any virtual machine that has been forced to disconnect will not be logged off.
- The ShutdownTime and Signal ShutdownTime features enable a virtual machine to register with CP to receive a shutdown signal when z/VM is shutting down (see "Steps for enabling Linux virtual servers to shut down automatically" on page 87). CP waits to shut itself down until the time interval (in seconds) is exceeded, or all of the virtual machines enabled for the signal shutdown have reported a successful shutdown. Some Linux distributions support this function, which allows Linux to shut down cleanly before z/VM shuts down.

Before you begin: You need to end CP's access to the primary parm disk. See "Steps for releasing the primary parm disk" on page 37.

Perform these steps to update the FEATURES statement:

1. Edit the SYSTEM CONFIG Z file. Type this command and press the Enter key:

```
xedit system config z
```

2. Find the line containing the text "Set_Privclass" in the section titled "Features Statement." At the XEDIT command line, type this command and press the Enter key:

```
====> /set_privclass
```

3. At the command line, type this command and press the Enter key:

```
====> i
```

4. Type this line (indent three spaces), then press the Enter key twice:

```
   Enable ,
```

   You might also change the comments for "Auto_Warm_IPL" and "Clear_TDisk".

5. Under "Passwords_on_Cmds ,":
   a. Change the three instances of "yes" to "no". You might also want to change the comments for these lines to reflect your change.

b. Move the cursor to the line containing the text “Logon no”. After the word “no” place a space and a comma.

6. Move the cursor to the prefix area on the line “Vdisk Userlim 144000 blocks ,”, type “i4” followed by a space, then press the Enter key.

7. Type these features (indent as shown) on the new lines:

```
Disconnect_timeout off
Set ,
  ShutdownTime 30 ,
  Signal ShutdownTime 500
```

8. Save the file. At the XEDIT command line, type this command and press the Enter key:

```
====> save
```

You should now see something like this:

```
SYSTEM   CONFIG   Z1  F 80  Trunc=80 Size=258 Line=148 Col=1 Alt=4

Features ,
  Disable ,                       /* Disable the following features */
    Set_Privclass ,               /* Disallow SET PRIVCLASS command */
  Enable ,
    Auto_Warm_IPL ,               /* Bypass prompting at IPL        */
    Clear_TDisk   ,               /* Clear TDisks when detached     */
  Retrieve ,                      /* Retrieve options               */
    Default  20 ,                 /* Default.... default is 20      */
    Maximum  255 ,                /* Maximum.... default is 255     */
  MaxUsers noLimit ,              /* No limit on number of users    */
  Passwords_on_Cmds ,             /* What commands allow passwords? */
    Autolog  no ,                 /* ... AUTOLOG does not           */
    Link     no ,                 /* ... LINK does not              */
    Logon    no ,                 /* ... and LOGON does not, too    */
  Vdisk Userlim 144000 blocks ,   /* Maximum vdisk allowed per user */
  Disconnect_timeout off
  Set,
    ShutdownTime 30 ,
    Signal ShutdownTime 500
```

Steps for controlling access to devices at startup

Sometimes your z/VM system may have access to devices that you do not want to be varied online during IPL. For instance, the devices may duplicate labels of devices used by your production system, or may be in use by other LPARs or systems. You can specify ranges of devices that z/VM should not vary online during IPL.

Before you begin: You need to end CP’s access to the primary parm disk. See “Steps for releasing the primary parm disk” on page 37.

Perform these steps to control access to devices at startup:

1. Edit the SYSTEM CONFIG Z file. Type this command and press the Enter key:

```
xedit system config z
```

2. Find the section titled “Status of Devices.” At the XEDIT command line, type this command and press the Enter key:

```
====> /status of devices
```

3. Find the line containing the text "Online_at_IPL." At the XEDIT command line, type this command and press the Enter key:

```
====> /online_at_ipl
```

4. At the command line, type this command and press the Enter key:

```
====> i
```

5. Type this line, then press the Enter key twice:

```
Offline_at_IPL rdevs,
```

where *rdevs* is a list of real device numbers.

Example:

```
Offline_at_IPL 0291–0592,
```

6. Save the file. At the XEDIT command line, type this command and press the Enter key:

```
====> save
```

You should now see something like this:

```
 SYSTEM   CONFIG   Z1  F 80  Trunc=80 Size=284 Line=210 Col=1 Alt=0

/*********************************************************************/
/*                        Status of Devices                          */
/*********************************************************************/

 Devices ,
   Online_at_IPL   0000-FFFF,
Offline_at_IPL 0291-0592,
   Sensed          0000-FFFF
```

Steps for defining a virtual switch

The recommended facility to use when virtual machines need to have network connectivity with each other is a z/VM virtual switch. You can define a virtual switch dynamically using the CP DEFINE VSWITCH command, but attempts by guests to use the virtual switch fail if you do not do this before guests try to use it. Such a situation is disruptive to the Linux boot process. To be sure the virtual switch is always available for your Linux guests, define a virtual switch in the SYSTEM CONFIG file.

For an overview on configuring TCP/IP for Linux, see "TCP/IP networking options for Linux" on page 30.

This procedure defines the virtual switch for z/VM. Subsequent topics explain how to set up TCP/IP and your virtual machines to use the virtual switch.

Before you begin: You need to end CP's access to the primary parm disk. See "Steps for releasing the primary parm disk" on page 37.

Perform these steps to define a virtual switch:

1. Edit the SYSTEM CONFIG Z file. Type this command and press the Enter key:

```
xedit system config z
```

2. Find the section titled "Status of Devices." At the XEDIT command line, type this command and press the Enter key:

```
====> /status of devices
```

3. Find the line containing the text "Sensed." At the XEDIT command line, type this command and press the Enter key:

```
====> /sensed
```

4. At the command line, type this command and press the Enter key:

```
====> i
```

5. Type these lines, then press the Enter key twice.

```
DEFINE VSWITCH switch_name RDEV rdev [PRIROUTER]
MODIFY VSWITCH switch_name GRANT userid
```

   where:

   switch_name
   is a name you give the virtual switch

   rdev
   is the real device address of your QDIO OSA-Express device

   userid
   is the user ID of a virtual machine for which you are granting access to the virtual switch.

   Note that the PRIROUTER (primary router) option is in brackets, indicating it is optional. The PRIROUTER option is required if any of your Linux virtual servers will act as a router to other networks. If you specify the PRIROUTER option, do not use the brackets.

   Example: If the switch name is VSWITCH1, the real device address of your OSA-Express device is BC0, and your virtual machine is LINUX0, add these lines:

```
DEFINE VSWITCH VSWITCH1 RDEV BC0
MODIFY VSWITCH VSWITCH1 GRANT LINUX0
```

6. Repeat the MODIFY VSWITCH command with the GRANT option for each user ID you wish to use the virtual switch.

7. Save the file. At the XEDIT command line, type this command and press the Enter key:

```
====> save
```

You should now see something like this:

```
 SYSTEM   CONFIG   Z1  F 80  Trunc=80 Size=286 Line=210 Col=1 Alt=0

/*********************************************************************/
/*                        Status of Devices                          */
/*********************************************************************/

 Devices ,
   Online_at_IPL   0000-FFFF,
Offline_at_IPL 0291-0592,
   Sensed          0000-FFFF
DEFINE VSWITCH VSWITCH1 RDEV BC0
MODIFY VSWITCH VSWITCH1 GRANT LINUX0
```

Steps for setting addresses for consoles

During the first IPL of your z/VM system, you needed to specify a load parameter so you could communicate with the Stand-Alone Program Loader (SAPL). The reason is the new z/VM system did not know which device address to use to display messages and prompts. The installation system includes default device addresses for use as the system operator console and emergency messages console, but these addresses rarely correspond to your production hardware configuration. So you will not need to use the SAPL each time you IPL z/VM, you need to supply the address of your IPL console and your emergency messages console on the Operator_Consoles statement.

During IPL, CP tries each device on the Operator_Consoles statement (from left to right) until it finds an active device. If no devices on the list are active, CP loads a disabled wait state and terminates.

The emergency message console is used as an additional console during failures. Define the emergency console with the Emergency_Message_Console statement.

Before you begin: You need to end CP's access to the primary parm disk. See "Steps for releasing the primary parm disk" on page 37. You need to know the real device addresses for the operator console and the emergency message console.

Perform these steps to set up addresses for consoles:

1. Edit the SYSTEM CONFIG Z file. Type this command and press the Enter key:

```
xedit system config z
```

2. Find the Operator_Consoles statement. At the XEDIT command line, type this command and press the Enter key:

```
====> /operator_consoles
```

3. Replace all device addresses on the Operator_Consoles statement with the real device addresses for your operator consoles.

Notes:

a. You can have one or more console addresses.

b. If you are not using 3270 devices on your system and are using 3270 integrated consoles instead, the keyword "System_3270" designates a full-screen integrated 3270 console (also known as SYSG) and the keyword "System_Console" designates an integrated line mode 3270 console (also known as SYSC).

4. Use the same addresses for the Emergency_Message_Console statement as you did for the Operator_Consoles statement.

5. Save the file. At the XEDIT command line, type this command and press the Enter key:

```
====> save
```

You should now see something like this:

```
 SYSTEM   CONFIG   Z1  F 80  Trunc=80 Size=286 Line=222 Col=1 Alt=0

/*********************************************************************/
/*                    Console Definitions                            */
/*********************************************************************/

 Operator_Consoles     0021 System_3270 System_Console
 Emergency_Message_Consoles   0021 System_3270 System_Console
```

Steps for updating special escape character defaults

z/VM provides special escape characters that, when included in terminal input streams, tell CP to perform an action before evaluating subsequent characters in the stream. An important escape character is the line end character; this character, followed by "cp" indicates that what follows is a CP command and should not be given to Linux. The default line end character is "#".

Example: Issuing:

```
#cp query time
```

sends the QUERY TIME command to CP rather than Linux.

Unfortunately, the default line end character is "#", which is also a special character for Linux. For example, the prolog to a shell script usually has a character string known as the "shebang" consisting of the characters "#!" followed by the path to the shell program used to evaluate the shell script. If you enter those characters on a virtual machine console (for example, while using the line mode editor `ed`), CP strips off the # symbol and attempts to evaluate the characters as command input.

You can change the default special escape characters for each virtual machine individually, or set a system-wide default. If the primary purpose of your z/VM system is for Linux virtual servers, it makes sense to change the default system-wide. Chapter 10, "Performing run-time tasks," on page 97 shows you how to change the special escape characters for an individual virtual machine.

Before you begin: You need to end CP's access to the primary parm disk. See "Steps for releasing the primary parm disk" on page 37.

Perform these steps to update system-wide special escape character defaults:

1. Edit the SYSTEM CONFIG Z file. Type this command and press the Enter key:

```
xedit system config z
```

2. Find the section entitled "Special characters for system set here." At the XEDIT command line, type this command and press the Enter key:

```
====> /special characters for system set here
```

3. Move the cursor to the line with the text "Escape", then move it to the first quote mark.
4. Type "OFF".

```
Char_Delete   OFF     ,       /* System default ... @   */
Escape        OFF     ,       /* System default ... "   */
Line_Delete   OFF     ,       /* Default is cent sign   */
```

5. Move the cursor to "Line_End", and replace "#" with "%".

```
Line_End     '%'   ,       /* System default ... #   */
```

6. Save the file. At the XEDIT command line, type this command and press the Enter key:

```
====> save
```

You should now see something like this:

```
 SYSTEM   CONFIG   Z1  F 80  Trunc=80 Size=286 Line=233 Col=1 Alt=0

/*********************************************************************/
/*               Special characters for system set here              */
/*********************************************************************/

 Character_Defaults ,
   Char_Delete  OFF   ,       /* System default ... @   */
   Escape       OFF   ,       /* System default ... "   */
   Line_Delete  OFF   ,       /* Default is cent sign   */
   Line_End     '%'   ,       /* System default ... #   */
   Tab          OFF           /* System default ... "   */
```

Steps for checking the syntax of the SYSTEM CONFIG file

Be sure the SYSTEM CONFIG file has the correct syntax by using the CPSYNTAX command.

Before you begin: You need to end CP's access to the primary parm disk. See "Steps for releasing the primary parm disk" on page 37.

Perform these steps to check the syntax of the SYSTEM CONFIG file:

1. Access the 193 disk as X. From the command line, type this command and press the Enter key:

```
access 193 x
```

2. If you are still editing SYSTEM CONFIG, exit the XEDIT session. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

3. Run the CPSYNTAX command. From the command line, type this command and press the Enter key:

```
cpsyntax system config
Configuration file processing complete -- no errors encountered.
Ready;
```

4. Check for a zero return code. If you do not get a zero return code, modify the SYSTEM CONFIG file.

You are done when you get a zero return code from CPSYNTAX.

Steps for restoring CP's access to the primary parm disk

You need to configure DirMaint and TCP/IP, so do not shut down z/VM at this point. Because you are not shutting down, you need to restore CP's access to the parm disk.

Before you begin: You need to log on as MAINT.

Perform these steps to restore the primary parm disk:

1. Release MAINT's access to the parm disk.

```
release z
Ready;
```

2. Change MAINT's read/write access to read-only.

```
link * cf1 cf1 rr
Ready;
```

3. Restore CP's access to the parm disk.

```
cpaccess maint cf1 a sr
```

You should see responses like:

```
cpaccess maint cf1 a sr
CPACCESS request for mode A scheduled.
Ready;
CPACCESS request for MAINT's 0cf1 in mode A completed.
```

Chapter 4. Configuring the Directory Maintenance Facility

This topic describes how to configure the Directory Maintenance Facility (DirMaint). For an overview of DirMaint and user management, see "Planning for user management" on page 32.

If you need to repeat these tasks, be sure to shut DirMaint down.

Steps for enabling DirMaint

Before you begin: You need to have a license for the DirMaint product. For ordering information, see the announcement letter for z/VM or contact your IBM representative.

Perform these steps to enable DirMaint:

1. Log onto MAINT.
2. Check that MAINT has write access to minidisk 51D.
   a. From the command line, type this command and press the Enter key:

```
query accessed
Mode  Stat     Files  Vdev  Label/Directory
A      R/W         9  191   MNT191
B      R/W       130  5E5   MNT5E5
C      R/W        59  2CC   MNT2CC
D      R/W       185  51D   MNT51D
S      R/O       689  190   MNT190
Y/S    R/O      1008  19E   MNT19E
Ready;
```

   b. If 51D is not accessed as R/W, type this command and press the Enter key:

```
link maint 51d 51d m
Ready;
```

   Note: If MAINT cannot access 51d in R/W mode, another user has the link as R/W and must change its access to R/O. **Do not use** *mw* mode for MAINT's link.

3. Access MAINT's 51D. From the command line, type this command and press the Enter key:

```
access 51d d
Ready;
```

4. Enable DirMaint. From the command line, type this command and press the Enter key:

```
service dirm enable
```

   Result: The command:
   - Sets DirMaint as ENABLED within CP.
   - Updates the SYSTEM CONFIG file on the primary parm disk.

Continue to the next steps.

Steps for changing the passwords for DirMaint service machines

As shipped with z/VM, the DIRMAINT, DATAMOVE, and 6VMDIR10 virtual machines have directory entries, but with "NOLOG" instead of passwords, which prevents the virtual machine from being logged on. You must change the directory entries to include passwords.

Before you begin: You must log onto MAINT.

Perform these steps to update DIRMAINT's password:

1. Edit the USER DIRECT file.

```
xedit user direct c
```

2. Find the directory entry for DIRMAINT. On the XEDIT command line, type this command and press the Enter key:

```
====> /USER DIRMAINT
```

3. Type your password over the word "NOLOG". If you need to insert characters, press the Insert key. **Example:**

```
USER DIRMAINT MYPASSWD 32M 64M BDG
⋮
```

4. Repeat steps 2 and 3 for the DATAMOVE user.
5. Similarly, change the default password for the 6VMDIR10 user.
6. Check 6VMDIR10 for this statement:

```
⋮
LINK MAINT 2CC 2CC RR
⋮
```

   If the statement does not exist, add it to the LINK statements for this user.

7. Save the file. From the XEDIT command line, type this command and press the Enter key:

```
====>  file
```

8. Place the new user directory online. From CMS, type this command and press the Enter key:

```
directxa user direct c
z/VM USER DIRECTORY CREATION PROGRAM - VERSION 5 RELEASE 1.0
EOJ DIRECTORY UPDATED AND ON LINE
Ready; T=0.13/0.15 14:33:34
```

9. Disconnect from MAINT.

   Tip: Before you disconnect from a virtual machine like MAINT, issue the CP command SET RUN ON. This command causes any programs running in the virtual machine to continue running even if the virtual machine enters a CP

READ state. You can place `CP SET RUN ON` in MAINT's PROFILE EXEC to avoid having to execute the command every time you disconnect from MAINT.

```
set run on
disc
```

Continue to the next steps.

Steps for configuring DirMaint

6VMDIR10 is the user ID that does service and maintenance for DirMaint. In keeping with the practice for products maintained by the VMSES/E component of z/VM, the virtual machine user ID is the same as the product identifier. In this procedure, you use 6VMDIR10 to configure DirMaint.

Before you begin: Make sure the DIRMAINT virtual machine is logged off. You need to log onto the 6VMDIR10 virtual machine.

Perform these steps to perform an initial configuration:

1. Access the 492 minidisk:

   ```
   acc 492 u
   ```

2. Issue this command:

   ```
   dir2prod samp 6vmdir10 dirm
   .
   .
   .
   DIR2PROD: Copy of 2C2 samples to 1DF disk has completed.
   DIR2PROD: Normal Termination.
   Ready; T=0.13/0.19 07:47:30
   ```

3. Issue this command:

   ```
   dir2prod access_new 6vmdir10 dirm
   .
   .
   .
   DMSACP726I 492 u releases
   DIR2PROD: Normal Termination
   Ready;
   ```

4. Access the 11F disk a z. From the command line, type this command and press the Enter key:

   ```
   access 11f z
   Ready;
   ```

5. Create a file called CONFIGAA DATADVH Z. From the command line, type this command and press the Enter key:

   ```
   xedit configaa datadvh z
   ```

   Tip: Files with the naming convention CONFIG*nn* DATADVH (*nn* can be alphabetic or numeric) provide DirMaint override instructions. If there are multiple CONFIG* DATADVH files, all are processed in reverse EBCDIC

order (CONFIG99 before CONFIG0 before CONFIGZ9 before CONFIGA, before CONFIG). For now, keep it simple by creating only one, CONFIGAA DATADVH.

6. At the XEDIT command line, type this command and press the Enter key:

```
====>  input
```

7. Add these statements to the file:

```
 CONFIGAA DATADVH  Z2  V 80  Trunc=80 Size=5 Line=1 Col=1 Alt=0
====>
      |...+....1....+....2....+....3....+....4....+....5....+....6....+....7...
    0 * * * Top of File * * *
    1    ALLOW_ASUSER_NOPASS_FROM=            VSMSERVE *
    2    DATAMOVE_MACHINE=                    DATAMOVE vmnetid  *
    3    DISK_CLEANUP=                        YES
    4    ONLINE=                              IMMED
    5    RUNMODE=                             OPERATIONAL
    6 * * * End of File * * *
```

where *vmnetid* is the VM network identifier. The network identifier is the same as the system identifier you defined in "Steps for updating the default system identifier" on page 39.

8. To exit input mode, press the Enter key twice.

9. Save the file. From the command line, type this command and press the Enter key:

```
====>  file
```

10. Release the 11F disk. From the command line, type this command and press the Enter key:

```
rel z
Ready;
```

Continue with the next steps.

Steps for authorizing users to perform DirMaint tasks

The AUTHFOR CONTROL file specifies user IDs allowed to perform authorized DirMaint tasks. It would be convenient to specify certain user IDs to do these tasks; otherwise, you need to log on the DIRMAINT virtual machine each time you want to do them. In this procedure, you authorize MAINT to perform DirMaint tasks.

Before you begin: You need to log onto the 6VMDIR10 virtual machine.

Perform these steps to create the authorization file:

1. Create and edit the AUTHFOR CONTROL file on the J-disk.

```
xedit authfor control j
```

2. On the XEDIT command line, type this command and press the Enter key to enter input mode:

```
====>  input
```

3. Type these lines in the input area:

```
⋮
ALL MAINT * 140A ADGHOPS
ALL MAINT * 150A ADGHOPS
```

4. Press the Enter key twice to return to editing mode.
5. From the XEDIT command line, type this command and press the Enter key:

```
====>  file
```

Continue to the next steps.

Steps for controlling where DirMaint creates minidisks

The EXTENT CONTROL file tells DirMaint where to allocate minidisks. The file is divided into sections. The first section identifies DASD extents on individual devices that can be used for minidisk allocation. The next section allows you to group devices by name and refer to them by a single name. For example, you can create a group called SYSTEM for minidisks related to system user IDs, or a group called LINUX for minidisks related to virtual machines for Linux. Grouping allows you to separate user minidisks from system-related minidisks.

Before you begin: Make sure the DIRMAINT virtual machine is logged off. You need to log onto the 6VMDIR10 virtual machine.

Perform these steps to control where DirMaint creates minidisks:

1. Edit the EXTENT CONTROL file on the J-disk.

```
xedit extent control j
```

2. Find the section labelled ":REGIONS". From the XEDIT command line, type this command and press the Enter key:

```
====> /:regions
```

3. Move the cursor to the prefix area for the line containing "*RegionID". Type "i" in the prefix area and press the Enter key:

```
00034 * ***************************************************
00035 :REGIONS.
i       *RegionId     VolSer     RegStart   RegEnd       Type
```

4. Move the cursor to the body of the blank line and type a statement with this pattern:

```
RegionId VolSer RegStart RegEnd Type
```

where

RegionId
Is the name of the region. Region names must be unique. If a region entry shares the same name with another region entry, the first record is used, the second entry is ignored. Region names may consist of the characters A-Z, 0-9, #, @, and $. DirMaint requires that this field be eight characters or less.

VolSer
Is the volume label of the real DASD. This value represents the value placed into volser field on any minidisks generated from this region. Volume IDs supported by DirMaint consist of a maximum of six characters A-Z, 0-9, #, @, and $. *VolSer* must be unique.

RegStart
Is the starting cylinder for this region.

RegEnd
Is the ending cylinder for this region.

Type
Is the device type of the DASD.

Example: This statement labels an entire 3390-03 with a volume label 430LIN as LINUX01:

```
LINUX01 430LIN 001 END 3390-03
```

(END is a special keyword specifying the last cylinder.)

5. Repeat step 4 on page 55 for all the volumes you need for your Linux virtual servers.

   Note: The region names must be unique.

6. Immediately after the volume statements for Linux virtual servers, add this statement for creating minidisks for other virtual machines.

   ```
   610W02  610W02 1 END devtype
   ```

   where *devtype* is the device type of volume 610W02.

7. Move the cursor to the prefix area for the line containing "*GroupName". Type "i2" in the prefix area and press the Enter key:

   ```
   00121 :GROUPS.
   i2      *GroupName RegionList
   ```

8. Type these two lines:

   ```
   GRPLNX (ALLOCATE ROTATING)
   GRPLNX LINUX01
   ```

   If you have additional regions, add those region IDs to the second line.

   Example:

   ```
   GRPLNX LINUX01 LINUX02 LINUX03
   ```

9. Move the cursor to the prefix area for the line with "* USERID ADDRESS", type "i2" in the prefix area, then press the Enter key:

```
00139 :EXCLUDE.
i2    * USERID ADDRESS
00141 :END.
```

10. Type these two lines:

```
MAINT 012*
SYSDUMP1 012*
```

11. Save the file. From the XEDIT command line, type this command and press the Enter key:

```
====>  file
```

Your file should look similar to this:

```
 EXTENT   CONTROL  J2  V 254  Trunc=254 Size=89 Line=35 Col=1 Alt=0
====>
   35 :REGIONS.
   36 *RegionId  VolSer    RegStart      RegEnd  Dev-Type  Comments
⋮
   46 LINUX01   430LIN           1         END  3390-03
⋮
   57 610W02    610W02           1         END  3390-03
   58 :END.
   59 :GROUPS.
   60 *GroupName RegionList
   61 DVHUSR    (ALLOCATE ROTATING)
⋮
   65 GRPLNX    (ALLOCATE ROTATING)
   66 GRPLNX    LINUX01 LINUX02 LINUX03
⋮
   71 :END.
   72 :EXCLUDE.
   73 * UserId Address
   74   MAINT    012*
   75   SYSDUMP1 012*
   76 :END.
⋮
```

Continue to the next steps.

Steps for copying the current USER DIRECT file

Before DirMaint is activated, the current USER DIRECT source file on MAINT's 2CC matches the online user directory. By copying the current USER DIRECT source file from MAINT's 2CC to the 6VMDIR10's J-disk, you create a starting point for DirMaint. From the USER DIRECT file, DirMaint processes the statements to create its own internal working structure for the user directory.

Important: After performing these steps, do not use the USER DIRECT file on MAINT's 2CC. To remind you that you must not use the file, you could rename it to USER NO-USE or USERDRCT UPBYDIRM.

Before you begin: You need to log onto 6VMDIR10.

Perform these steps to copy the current USER DIRECT file:

1. Link to MAINT's 2CC minidisk:

```
link maint 2cc 2cc mr multiple
```

2. Access the 2CC minidisk (MAINT's 2CC, which is linked to 6VMDIR10):

```
access 2cc z
```

3. Copy MAINT's USER DIRECT file to 6VMDIR10's J-disk as USER INPUT:

```
copy user direct z user input j
```

4. Release the linked minidisk:

```
release z (detach
```

Continue with the next steps.

Steps for putting the configuration into production and starting DirMaint

In these steps, you place the DirMaint configuration into production and start DirMaint.

Before you begin: You need to log onto 6VMDIR10. Later, when you log onto DIRMAINT, you need to know the log-on password you defined in "Steps for changing the passwords for DirMaint service machines" on page 52.

Perform these steps to put the configuration into production and start DirMaint:

1. From the CMS command line, type this command and press the Enter key:

```
dir2prod prod 6vmdir10 dirm
```

2. Log off 6VMDIR10. Type this command and press the Enter key:

```
logoff
```

3. Log onto DIRMAINT.
4. Start the DirMaint program:

```
dvhbegin
⋮
DVHWAI2140I Waiting for work on 04/02/09 at 18:47:38.
```

5. Disconnect DIRMAINT. Type this command and press the Enter key:

```
cp disc
```

Continue to the next steps.

Steps for automatically starting DIRMAINT

In this procedure, you update AUTOLOG1's PROFILE EXEC to automatically log on (autolog) DIRMAINT when z/VM starts.

Before you begin: You need to log onto MAINT.

Perform these steps to have DIRMAINT automatically logged on:

1. Link to the AUTOLOG1 191 minidisk. Type this command and press the Enter key:

```
link autolog1 191 091 mr
Ready;
```

2. Access the 091 minidisk as Z. Type this command and press the Enter key:

```
access 091 z
Ready;
```

3. Edit AUTOLOG1's PROFILE EXEC. Type this command and press the Enter key:

```
xedit profile exec z
```

4. Go to the bottom of the file. From the XEDIT command line, type this command and press the Enter key:

```
====> bot
```

5. Add the new XAUTOLOG statements. From the XEDIT command line, type "input", press the Enter key, then type these lines. Press the Enter key after you type each line:

```
ADDRESS COMMAND CP XAUTOLOG DIRMAINT
ADDRESS COMMAND CP XAUTOLOG DATAMOVE
ADDRESS COMMAND CP LOGOFF
```

6. Press the Enter key.

   Result: The file should look like this:

```
 PROFILE EXEC     Z2  V 80  Trunc=80 Size=7 Line=4 Col=1 Alt=1

* * * Top of File * * *
/**************************/
/*  Autolog1 Profile Exec  */
/**************************/
ADDRESS COMMAND CP AUTOLOG VMSERVS VMSERVS
ADDRESS COMMAND CP AUTOLOG VMSERVU VMSERVU
ADDRESS COMMAND CP AUTOLOG VMSERVR VMSERVR
ADDRESS COMMAND CP AUTOLOG DTCVSW1
ADDRESS COMMAND CP AUTOLOG DTCVSW2
ADDRESS COMMAND CP XAUTOLOG DIRMAINT
ADDRESS COMMAND CP XAUTOLOG DATAMOVE
ADDRESS COMMAND CP LOGOFF
```

7. Save the file. At the XEDIT command line, type this command and press the Enter key:

```
====> file
```

8. Detach the 091 minidisk (AUTOLOG1's 191). Type this command and press the Enter key:

```
release z (det
Ready;
```

You are done.

Steps for testing DirMaint

This procedure tests DirMaint. The command you use allows MAINT to issue DirMaint commands without a password.

Before you begin: You need to know the password for MAINT.

Perform these steps to test DirMaint:

1. Log onto MAINT.
2. From the CMS command line, type this command and press the Enter key:

```
dirm needpass no
```

3. When prompted, type MAINT's password and press the Enter key.

You see these messages:

```
dirm needpass no
DVHXMT1181R Enter the current logon password of MAINT at DVHTEST6 for
DVHXMT1181R authentication.  It will not be displayed on the
DVHXMT1181R terminal.  To exit without processing the command, just
DVHXMT1181R press ENTER.

DVHXMT1191I Your NEEDPASS request has been sent for processing.
MAINT AT DVHTEST6; T=0.05/0.06 18:53:02
 DVHREQ2288I Your USEROPTN request for MAINT at * has been accepted.
 DVHBIU3450I The source for directory entry MAINT has been updated.
 DVHBIU3456I Object directory update is not required for this source
 DVHBIU3456I update.
 DVHREQ2289I Your USEROPTN request for MAINT at * has completed;
 DVHREQ2289I with RC = 0.
```

You are done.

Step for modifying the OPERATOR's directory entry

In Chapter 9, "Setting up basic system automation," on page 85, you set up the programmable operator facility, which requires that the OPERATOR virtual machine automatically load (IPL) CMS when it is logged on. This procedure uses DirMaint to modify the OPERATOR's directory entry to include a statement that loads CMS.

Before you begin: You need to be logged onto MAINT.

Perform this step to modify the OPERATOR's directory entry:

- From the command line, type this command and press the Enter key:

```
dirm for operator ipl cms
DVHXMT1191I Your IPL request has been sent for processing.
Ready;
 DVHREQ2288I Your IPL request for OPERATOR at * has been accepted.
 DVHBIU3450I The source for directory entry OPERATOR has been updated.
 DVHBIU3423I The next ONLINE will take place via Diagnose 84.
 DVHBIU3428I Changes made to directory entry OPERATOR have been placed
 DVHBIU3428I online.
 DVHBIU3427I Changes made to directory entry OPERATOR by MAINT at
 DVHBIU3427I ZVMLINUX have been placed online.
 DVHREQ2289I Your IPL request for OPERATOR at * has completed; with RC
 DVHREQ2289I = 0.
```

You know you are done when you see a zero return code from DirMaint.

Chapter 5. Configuring TCP/IP

In this topic, you configure TCP/IP, if you have not already done so, and configure TCP/IP to start automatically whenever z/VM is initialized.

For an overview on configuring TCP/IP for Linux, see "TCP/IP networking options for Linux" on page 30.

Setting up the production TCP/IP

During z/VM installation, you had the option to set up a static connection to the TCP/IP network through the IPWIZARD. Such a configuration is suitable for installations that employ only static (as opposed to dynamic) network routes. If you want a more comprehensive TCP/IP network, you need to follow the instructions in *z/VM: TCP/IP Planning and Customization*, SC24-6238.

Table 5 outlines the tasks you must do:

Table 5. Task roadmap for the production TCP/IP

Subtask	Associated instructions (see . . .)
Setting up your production TCP/IP	If you set up a static connection during z/VM installation and are satisfied with this configuration, this task is optional. If you want a more comprehensive TCP/IP network, you need to follow the instructions in *z/VM: TCP/IP Planning and Customization*, SC24-6238.
Setting up TCP/IP to be automatically started	"Steps for automatically starting TCP/IP"

Steps for automatically starting TCP/IP

This procedure ensures the TCPIP virtual machine is always started when you IPL z/VM. Except for a few system user IDs, the AUTOLOG1 virtual machine automatically starts virtual machines at IPL time through its PROFILE EXEC. By placing an additional XAUTOLOG command for TCPIP in AUTOLOG1's PROFILE EXEC, that virtual machine is automatically logged on when you IPL z/VM.

Before you begin: You need to log on as MAINT.

Perform these steps to ensure TCP/IP is started at IPL time:

1. Link to the AUTOLOG1 191 minidisk. Type this command and press the Enter key:

```
link autolog1 191 091 mr
Ready;
```

2. Access the 091 minidisk as Z. Type this command and press the Enter key:

```
access 091 z
Ready;
```

3. Edit AUTOLOG1's PROFILE EXEC.
   a. Type this command and press the Enter key:

```
xedit profile exec z
```

   b. Go to the bottom of the file. From the XEDIT command line, type this command and press the Enter key:

```
====> bot
```

   c. Move up one line (The last line logs off AUTOLOG1. You need to execute the XAUTOLOG commands before logging off AUTOLOG1). From the XEDIT command line, type this command and press the Enter key:

```
====> up 1
```

   d. From the XEDIT command line, type this command and press the Enter key:

```
input
```

4. Add the new XAUTOLOG statement.

```
ADDRESS COMMAND CP XAUTOLOG TCPIP
```

5. Press the Enter key twice.

   Result: The file should look like this:

```
 PROFILE EXEC     Z2  V 80  Trunc=80 Size=7 Line=4 Col=1 Alt=1

* * * Top of File * * *
/**************************/
/*  Autolog1 Profile Exec  */
/**************************/
ADDRESS COMMAND CP AUTOLOG VMSERVS VMSERVS
ADDRESS COMMAND CP AUTOLOG VMSERVU VMSERVU
ADDRESS COMMAND CP AUTOLOG VMSERVR VMSERVR
ADDRESS COMMAND CP XAUTOLOG DTCVSW1
ADDRESS COMMAND CP XAUTOLOG DTCVSW2
ADDRESS COMMAND CP XAUTOLOG DIRMAINT
ADDRESS COMMAND CP XAUTOLOG DATAMOVE
ADDRESS COMMAND CP XAUTOLOG TCPIP
ADDRESS COMMAND CP LOGOFF
```

6. Save the file. At the XEDIT command line, type this command and press the Enter key:

```
====> file
```

7. Detach the 091 minidisk (AUTOLOG1's 191). Type this command and press the Enter key:

```
release z (det
Ready;
```

You are done.

Chapter 6. Restarting z/VM and checking the system

After making changes to the system, configuring DirMaint, and setting up the network configuration, restart z/VM and check that the changes you want were made. This topic explains how to do that.

Steps for restarting z/VM

In this procedure, you restart (IPL) z/VM and log onto MAINT, from which you can check that all your configuration changes are in effect.

Before you begin: You need to be logged on as MAINT. Be sure you have checked the syntax of the SYSTEM CONFIG file (see "Steps for checking the syntax of the SYSTEM CONFIG file" on page 48). Be sure other important z/VM processing is complete.

Perform these steps to restart z/VM:

1. Type this command and press the Enter key:

   ```
   shutdown reipl
   ```

   Result: When the system shuts down and re-IPLs, you see a number of IPL messages. z/VM restores the system to the same state as it was prior to shutdown (for instance, with OPERATOR disconnected).
2. To get a z/VM logo, press the Enter key.
3. Log onto MAINT.

You are done. Go on to the next procedure.

Steps for checking paging and spooling space

Before you begin: You need to be logged on as MAINT.

Perform these steps to check that paging and spooling space matches your intended definitions:

1. Type this command and press the Enter key. Check the response for paging volumes.

   ```
   query alloc page
              EXTENT EXTENT  TOTAL  PAGES   HIGH    %
   VOLID  RDEV  START    END  PAGES IN USE   PAGE USED
   ------ ---- ------ ------ ------ ------ ------ ----
   610PAG C331      0   3338 601020  13646   31175  2%
                             ------ ------         ----
   SUMMARY                   601020  13646           2%
   USABLE                    601020  13646           2%
   Ready; T=0.01/0.01 10:07:56
   ```

2. Type this command and press the Enter key. Check the response for spooling volumes.

```
query alloc spool
            EXTENT EXTENT  TOTAL  PAGES   HIGH    %
VOLID  RDEV  START    END  PAGES IN USE   PAGE USED
------ ---- ------ ------ ------ ------ ------ ----
610SPL C330      0   3338 601020   1964   8432   1%
                           ------ ------        ----
SUMMARY                   601020   1964          1%
USABLE                    601020   1964          1%
Ready; T=0.01/0.01 10:15:22
```

You are done. Go on to the next procedure.

Step for checking the system identifier

Before you begin: You need to be logged on as MAINT.

Perform this step to check the system identifier:

- Type this command and press the Enter key. Check the response for your system identifier.

```
identify
MAINT    AT system   VIA RSCS     02/20/04 20:11:10 UTC      FRIDAY
Ready;
```

where *system* is the system identifier.

Step for checking the user volume list

Before you begin: You need to be logged on as MAINT.

Perform this step to check the user volume list:

- Type this command and press the Enter key. Check for responses for the user volumes you defined previously. In the response, SYSTEM means the volume is a user volume.

```
query dasd
DASD 0190 CP SYSTEM CMS20    2
DASD AB24 CP OWNED  610RES  59
DASD AB25 CP OWNED  610W01  63
DASD C32F CP OWNED  610W02   1
DASD C330 CP OWNED  610SPL   0
DASD C331 CP OWNED  610PAG   0
DASD DB35 CP SYSTEM LNX001   0
DASD DB36 CP SYSTEM LNX002   0
DASD DB3F CP SYSTEM LNX003   0
DASD DB40 CP SYSTEM LNX004   0
Ready; T=0.01/0.01 10:19:29
```

Steps for checking features

Before you begin: You need to be logged on as MAINT.

Perform these steps to check features:

1. Check the shutdown time periods. Type these commands and press the Enter key after each command:

```
query shutdowntime
System shutdown time: 30 seconds; previous shutdown time: n seconds.
Ready;
query signal shutdowntime
System default shutdown signal timeout: 500 seconds
Ready;
```

2. Check the number of retrieve buffers. Type this command and press the Enter key:

```
query retrieve
10 buffers available.  Maximum of 20 buffers may be selected.
Ready;
```

Step for checking offline devices

Before you begin: You need to be logged on as MAINT.

Perform this step to check offline devices:

- Type this command and press the Enter key:

```
query dasd offline
DASD 1440 OFFLINE , DASD 1441 OFFLINE , DASD 1442 OFFLINE , DASD 1443 OFFLINE
```

Step for checking the virtual switch

Before you begin: You need to be logged on as MAINT.

Perform this step to check the virtual switch:

- Type this command and press the Enter key:

```
query vswitch details
VSWITCH SYSTEM VSWITCH1 Type: VSWITCH Connected: 0    Maxconn: INFINITE
   PERSISTENT  RESTRICTED    NONROUTER                 Accounting: OFF
   VLAN Unaware
   State: Ready
   IPTimeout: 5         QueueStorage: 8
   Portname: UNASSIGNED RDEV: 0BC0 Controller: DTCVSW1  VDEV:  0BC0
     VSWITCH Connection:
       RX Packets: 0          Discarded: 27         Errors: 0
       TX Packets: 0          Discarded: 0          Errors: 0
       RX Bytes: 0                    TX Bytes: 0
       Device: 0BC0  Unit: 000   Role: DATA
Ready;
```

The response shows the virtual switch is defined and ready for use ("State: Ready"), but no virtual machine is using it ("Connected: 0").

Step for checking character defaults

Before you begin: You need to be logged on as MAINT.

Perform this step to check character defaults:

- Type this command and press the Enter key:

```
query terminal
LINEND  % , LINEDEL OFF, CHARDEL OFF, ESCAPE OFF, TABCHAR  "
LINESIZE 080, ATTN OFF, APL OFF, TEXT OFF, MODE VM, HILIGHT OFF
CONMODE 3215, BREAKIN   IMMED , BRKKEY PA1  , SCRNSAVE OFF
AUTOCR ON , MORE 050 010, HOLD ON , TIMESTAMP OFF, SYS3270 OFF
Ready;
```

  The response should show the LINEND character is "%" and that the LINEDEL, CHARDEL, and ESCAPE characters are "OFF."

Steps for checking TCP/IP

Before you begin: You need to log on as TCPMAINT.

Perform these steps to check TCP/IP:

1. From the command line, type this command and press the Enter key:

```
ifconfig -a
name     inet addr: ip_address mask: mask
         UP BROADCAST MULTICAST MTU: 1492
         vdev: vdev rdev: rdev type: QDIO ETHERNET portname: UNASSIGNED
         ipv4 router type: NONROUTER ipv6: DISABLED
         cpu: 0 forwarding: ENABLED
         RX bytes: 127217 TX bytes: 126518
Ready;
```

   You defined the interface name (*name*), IP address (*ip_address*), mask (*mask*), virtual device address (*vdev*), and real device address (*rdev*) for your production TCP/IP through the IPWIZARD (see "Setting up the production TCP/IP" on page 63).

2. To check the network interface, ping the gateway. In the command, *ip_address* is the IP address of your gateway.

```
ping ip_address
Ping Level level 610: Pinging host ip_address.
                Enter 'HX' followed by 'BEGIN' to interrupt.
PING: Ping #1 response took 0.008 seconds. Successes so far 1.
Ready;
```

3. To check the network interface, ping an address outside your subnetwork. In the command, *ip_address* is an address outside your subnetwork.

```
ping ip_address
Ping Level level 610: Pinging host ip_address.
                Enter 'HX' followed by 'BEGIN' to interrupt.
PING: Ping #1 response took 0.008 seconds. Successes so far 1.
Ready;
```

You are done checking the system configuration.

Chapter 7. Creating your first Linux virtual machine and installing Linux

This topic covers configuring your first Linux virtual machine and installing your first Linux operating system.

Overview of defining virtual machines for Linux

Previous sections have shown you how to configure z/VM functions and facilities in order to create the system infrastructure for your virtual machines: configuring z/VM, enabling and configuring DirMaint, and configuring TCP/IP. This section and the next, Chapter 8, "Cloning Linux virtual servers," on page 83, show you how to install your first Linux operating system and then use a replication or cloning process to create additional Linux virtual servers quickly. The cloning process allows you to create a new Linux virtual server without the need to install the Linux operating system from scratch.

Before you begin defining Linux servers, read Chapter 2, "Planning for Linux virtual servers," on page 21.

The basic subtasks are:

Table 6. Task roadmap for setting up Linux virtual servers

Subtask	Associated instructions (see . . .)
Define a prototype and create your first virtual machine from the prototype	"Steps for defining a master virtual machine for Linux"
Install the Linux operating system in the virtual machine.	Follow the instructions for your Linux distribution. To get started, see "Installing Linux in a virtual machine" on page 77.
Replicate, or clone, additional Linux virtual servers through DirMaint prototype and CLONEDISK functions.	"Steps for cloning a Linux virtual server" on page 83

Steps for defining a master virtual machine for Linux

These steps tell you how to define a master virtual machine that uses the default 2.2G on a 3390-3 DASD.

In this procedure, you modify two sample files, LINDFLT DIRECT and LINUX PROTODIR. LINDFLT DIRECT is a shared profile for all Linux systems and defines common definitions for all your Linux virtual servers. LINUX PROTODIR is designed to define unique characteristics of a virtual machine, such as the DASD definitions.

Before you begin: Log on to MAINT.

Perform these steps to define the master virtual machine:

1. Get the LINDFLT and LINUX samples from DirMaint.
   a. From the command line, type these commands and press the Enter key:

      ```
      dirm send lindflt direct
      dirm send linux protodir
      ```

      Result: DirMaint sends the sample files to MAINT'S reader.
   b. If you have other files in your reader, queue the two files you get from DirMaint with the ORDER command.

      Example: DirMaint sent you LINDFLT DIRECT and LINUX PROTODIR with spool identifiers 0004 and 0005, and you already have spool files 0001, 0002, and 0003 in your reader. Issue this command to queue the two files from DirMaint:

      ```
      order rdr 4 5
      0000002 FILES ORDERED
      Ready;
      ```

   c. Receive the sample files. From the command line, type these commands and press the Enter key:

      ```
      receive
      File LINDFLT DIRECT A1 created from LINDFLT DATADVH A1 received from DIRMAINT at system
      receive
      File LINUX PROTODIR A1 created from LINUX DATADVH A1 received from DIRMAINT at system
      ```

2. Edit LINDFLT DIRECT. From the command line, type this command and press the Enter key:

   ```
   xedit lindflt direct a
   ```

   Result: Your LINDFLT DIRECT looks like:

   ```
   PROFILE LINDFLT
   CLASS G
   IPL CMS
   MACHINE ESA
   MAXSTORAGE 2047M
   OPTION QUICKDSP
   STORAGE 128M
   CONSOLE 0009 3215 T
   NICDEF 600 TYPE QDIO LAN SYSTEM VSWITCH1
   SPOOL 000C 2540 READER *
   SPOOL 000D 2540 PUNCH A
   SPOOL 000E 1403 A
   LINK MAINT 0190 0190 RR
   LINK MAINT 019D 019D RR
   LINK MAINT 019E 019E RR
   LINK TCPMAINT 0592 0592 RR
   ```

   Notes:
   a. If you log onto virtual machines cloned from LINDFLT DIRECT, the command IPL CMS places the virtual machine in VM READ state. You must press the Enter key to finish loading CMS.
   b. The name of the virtual switch must match the name you used when you defined the virtual switch for the system. See "Steps for defining a virtual switch" on page 44.

c. For all virtual machines that you clone using LINDFLT DIRECT, you cannot define other devices at virtual addresses 600, 601, and 602. All those virtual machines will have the same virtual device addresses for the virtual switch.

3. If you want to give all your cloned Linux virtual machines access to the cryptographic facility:

   a. Add a line before the OPTION statement:

```
00000 * * * Top of File * * *
00001 PROFILE LINDFLT
00002 CLASS G
00003 IPL CMS
00004 IUCV ALLOW
00005 MACHINE ESA
a     MAXSTORAGE 2047M
00007 OPTION QUICKDSP
00008 STORAGE 128M
```

   b. On the new line, type:

```
CRYPTO APVIRT
```

   For more information about the cryptographic facility, see "Giving Linux virtual servers access to cryptographic hardware for SSL acceleration" on page 31.

4. If you make any changes to LINDFLT DIRECT, save the file with the XEDIT FILE command. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

5. Edit LINUX PROTODIR.

   a. From the XEDIT command line, type this command and press the Enter key:

```
xedit linux protodir
```

   Result: Your LINUX PROTODIR looks like:

```
USER LINUX NOLOG
INCLUDE LINDFLT
MDISK 191 3390 AUTOG 0050 LINGROUP MR
MDISK 150 3390 AUTOG 3088 LINGROUP MR
MDISK 151 3390 AUTOG 0200 LINGROUP MR
```

   b. Change the minidisk size for MDISK 191 to 0005:

```
MDISK 191 3390 AUTOG 0005 LINGROUP MR
```

   c. Change each occurrence of "LINGROUP" to "GRPLNX":

```
MDISK 191 3390 AUTOG 0005 GRPLNX   MR
MDISK 150 3390 AUTOG 3088 GRPLNX   MR
MDISK 151 3390 AUTOG 0200 GRPLNX   MR
```

Notes:

a. For the swap disk 151, you could use a virtual disk, but be aware of the impact a virtual disk has on your system. See "Linux Performance when running under VM" (http://www.vm.ibm.com/perf/tips/linuxper.html).
b. "GRPLNX" matches the region name you created in "Steps for controlling where DirMaint creates minidisks" on page 55.

6. If you make any changes to LINUX PROTODIR, save the file with the XEDIT FILE command. Otherwise, quit the file: from the XEDIT command line, type this command and press the Enter key:

```
====> quit
```

7. Install the LINDFLT DIRECT and LINUX PROTODIR files.
   a. Start by testing for the existence of a user called LINDFLT. Issue:

```
dirm for lindflt get lock
```

      - If the GET LOCK command succeeded, issue:

```
dirm for lindflt replace
```

      - If the GET LOCK command did not succeed, issue:

```
dirm add lindflt
```

   b. Install LINUX PROTODIR. Issue:

```
dirm file linux protodir
```

8. Use DirMaint to define the master virtual machine (LINMSTR). Type this command and press the Enter key:

```
dirm add linmstr like linux pw new_password
```

   where *new_password* is the password for logging on LINMSTR. *new_password* must be 8 characters long and must contain both numbers and letters.

9. If you decide you want to delete a virtual machine you cloned, type this command and press the Enter key:

```
dirm for userid purge
```

   where *userid* is the user ID of the virtual machine you want to delete.

Guidelines:

- You might want to create additional prototypes (*fn* PROTODIR) for meeting differing processing requirements of your Linux virtual servers. You can use naming conventions that indicate the processing requirements satisfied by each prototype. For instance, you might want to create a small memory prototype (128 MB virtual machine called LINUX128), a medium memory prototype (256 MB virtual machine called LINUX256), and a large memory prototype (512 MB virtual machine called LINUX512), then create small, medium, and large master virtual machines, into which you can install the Linux operating system and appropriate application packages or middleware (the large master could have

WebSphere, for instance). Then, during the cloning process, you can select the master prototype that meets the needs of your clone.

- The LINUX PROTODIR consumes an entire 3390-3 volume. To help slow the consumption of 3390-3 volumes, you can split the Linux file system into read-only and read/write portions and share the read-only portion. You should be familiar with Linux file system usage practices before attempting to set up read-only portions of the file system. To do this, you need to create a master Linux virtual server that can update the minidisk to which other Linux virtual servers are linked in read-only mode through the LINK command or LINK directory statement.

  Note: There are important maintenance issues here: running Linux virtual servers do not get updates to the read-only disk until those servers are recycled. Also, a running Linux system linked to read-only disks might crash if those disks are updated while it is running. So, you need to develop a maintenance plan to determine how to update the read-only disk and when to recycle Linux virtual servers.
- If you want to see the entire user directory entry for a user, issue the following commands:

```
dirm for user_id get nolock
...
RDR FILE nnn SENT FROM DIRMAINT PUN WAS 8116 RECS 0027 CPY  001 A NOHOLD NOKEEP
...
peek nnn (for *
```

  where *user_id* is the user ID you want to see and *nnn* is the reader spool file identifier.

You know you are done when DirMaint tells you LINMSTR is created.

Steps for setting up LINMSTR's disks

In this procedure, you format LINMSTR's 191 disk and create a PROFILE EXEC. Each clone you create gets a replica of this 191 disk.

Because the 191 disk has only five cylinders, it is too small to hold the Linux boot files for installing the Linux operating system. It would also be a waste of disk space to increase the size of the 191 disk to accommodate the Linux boot files, since each clone would get the same sized disk and the clones do not need a larger disk. Instead, this procedure has you create a 192 disk for LINMSTR only. The larger 192 disk can hold the Linux boot files and this disk will not be replicated in the clones.

Before you begin: You need to log on to MAINT. Later in this procedure, you need to log onto LINMSTR.

Perform these steps to set up LINMSTR's disks:

1. Log onto MAINT and from the command line, type this command and press the Enter key:

```
dirm for linmstr amdisk 192 devtype autov 50 610W02
DVHXMT1191I Your AMDISK request has been sent for processing.
Ready;
 DVHREQ2288I Your AMDISK request for LINMSTR at * has been accepted.
 DVHSCU3541I Work unit 18124944 has been built and queued for processing.
 DVHSHN3541I Processing work unit 18124944 as MAINT from ZVMLINUX,
 DVHSHN3541I notifying MAINT at ZVMLINUX, request 62 for LINMSTR sysaffin
 DVHSHN3541I *; to: AMDISK 0192 devtype AUTOV 50 610W02
 DVHBIU3450I The source for directory entry LINMSTR has been updated.
 DVHDRC3428I Changes made to directory entry LINMSTR have just been
 DVHDRC3428I placed online.
 DVHSHN3430I AMDISK operation for LINMSTR address 0192 has finished (WUCF
 DVHSHN3430I 18124944).
 DVHREQ2289I Your AMDISK request for LINMSTR at * has completed; with
 DVHREQ2289I RC = 0.
```

   where *devtype* is the disk device type of volume 610W02 (for instance, 3390).

2. Log onto LINMSTR.

   Note: After the logon messages, you receive a message because the 191 disk is not formatted. Disregard this message:

```
DMSACP112S A(191) device error
```

3. Format the 191 disk with the CMS FORMAT command. From the command line, type these commands and press the Enter key:

```
format 191 a
DMSFOR603R FORMAT will erase all files on disk A(191). Do you wish to continue?
Enter 1 (YES) or 0 (NO).
1
DMSFOR605R Enter disk label:
lin191
DMSFOR733I Formatting disk A
DMSFOR732I 5 cylinders formatted on Z(191)
Ready;
```

4. Create a PROFILE EXEC on the 191 disk.

   a. Issue this command:

```
xedit profile exec a
```

   b. On the XEDIT command line, type this command and press the Enter key:

```
====> input
```

   c. Type in these statements, then press the Enter key twice.

      Rule: The first line of any PROFILE EXEC must be a comment. Comments are bounded by "/*" and "*/".

```
/*  Sample PROFILE EXEC for Linux servers */
"CP TERM LINEND %"      /* Use %CP to talk to CP    */
"CP SET PF12 RETRIEVE"  /* Recall previous commands */
"CP TERM MORE 1 0"      /* Clear screen after 1 sec */
"CP SET RUN ON"         /* CP READ won't stop server*/
"CP TERM HOLD OFF"      /* (Look in book)           */
"ACCESS 592 Z"          /* Get to TCP/IP functions  */
```

Note: Add 'CP TERM LINEND %' in case the system default line end character is changed.

Guideline: In Chapter 9, "Setting up basic system automation," on page 85, you automate the Linux virtual console, which requires a modification of the PROFILE EXEC for each Linux virtual server you want to automate. If you prefer to have this function part of every Linux clone, you can add the required statement now. See "Steps for automating Linux virtual consoles" on page 93.

d. Save the file. On the XEDIT command line, type this command and press the Enter key:

```
====> file
```

5. Format the 192 disk with the CMS FORMAT command. From the command line, type these commands and press the Enter key:

```
format 192 a
DMSFOR603R FORMAT will erase all files on disk A(192). Do you wish to continue?
Enter 1 (YES) or 0 (NO).
1
DMSFOR605R Enter disk label:
lin192
DMSFOR733I Formatting disk A
DMSFOR732I 50 cylinders formatted on A(192)
Ready;
```

You are done when you finish formatting the 192 disk.

Installing Linux in a virtual machine

Though you need to follow the instructions from your Linux distribution to install and configure Linux, and those instructions differ from distribution to distribution, virtual machine facilities and tasks common to all distributions are covered in this section.

Overview of installing Linux in a virtual machine

You must follow your distribution documentation to install the Linux operating system in your new LINMSTR virtual machine, which you created in "Steps for defining a master virtual machine for Linux" on page 71. For information about obtaining your Linux distribution, see "Steps for obtaining documentation and media" on page 34. This topic includes information about your virtual machine configuration that you should know for your Linux distribution.

To get the installation and configuration process started, Linux distributions follow this pattern:

Stage	Description
1	Use FTP to move the Linux boot files to a target virtual machine. As of this writing, popular Linux distributions require you to ftp the files to your virtual machine. **Examples:** SUSE Linux uses these boot files: • VMRDR IKR. The Linux kernel built for use with a ram disk. • INITRD. The initial ram disk image. • PARM FILE. The generic parm file for use with the ram disk system. Red Hat, Inc., Linux uses these boot files: • KERNEL IMG. The Linux kernel. • INITRD IMG. The initial ram disk image. • REDHAT PARM. The generic parm file for use with the ram disk system. **Notes:** 1. You must ftp the Linux kernel and ram disk image files to the virtual machine by specifying them as binary files with a VM record format of fixed length 80. It is recommended that you use the z/VM FTP command to transfer the files. If you use the FTP command, use the commands "bin" and "locsite fix 80" before you make the transfer. If you use the workstation FTP client, issue the commands "bin" and "quote site fixrecfm 80" before you make the transfer. See "Example of using FTP to get the Linux boot files" on page 79. 2. If the hardware management console (HMC) supports accessing HMC removable media from the z/VM logical partition, you do not need a remote FTP server to get the Linux installation files. See "Example of using FTP to install Linux from the hardware management console," on page 139.
2	Punch the Linux boot files to the virtual machine reader. See "Example of punching Linux boot files to the virtual machine reader" on page 80.
3	Boot (IPL) Linux from the virtual machine reader. See "Example of booting (IPL) the Linux boot files from the virtual machine reader" on page 81.
4	Use the tools and documentation from your Linux distribution to install and configure Linux. **Note:** Installation of Linux itself, not just the transfer of the installation files, can also be accomplished using FTP.

During Linux configuration, you need to know these important points about the LINMSTR virtual machine configuration:

- When you log on to LINMSTR, CMS is loaded automatically and the PROFILE EXEC is automatically executed. This allows you to interact with z/VM to get the Linux installation process started.
- Load the Linux boot files onto the **192** disk. The 191 disk is too small to hold the boot files. You must access 192 as your A-disk:

```
access 192 a
DMSACC724I 192 replaces A (191)
Ready;
```

- The default memory for LINMSTR is 128 MB.
- The 150 minidisk is for the Linux root file system.
- The 151 minidisk is for the Linux swap space.

- LINMSTR connects to the IP network through the virtual network interface connection that you defined in LINDFLT PROTODIR. There are three qeth device addresses that start at the virtual address that you defined on the NICDEF statement.

  Example: If you used the statement:

  ```
  NICDEF 600 TYPE QDIO LAN SYSTEM VSWITCH1
  ```

  the qeth device addresses you enter during Linux configuration are 0.0.0600, 0.0.0601, and 0.0.0602.

  For information about device drivers, see *Linux on System z: Device Drivers, Features, and Commands* on the IBM developerWorks Linux on System z Web site entitled "Documentation for Development stream" at:

  http://www.ibm.com/developerworks/linux/linux390/documentation_dev.html

Guidelines:

- Before you install Linux, format the 150 and 151 minidisks. From the CMS command line, type these commands, press the Enter key, then answer the prompts:

  ```
  format 150 z
  ⋮
  format 151 x
  ```

- During the Linux installation process, predefine extra device addresses, making it unnecessary to do this after the system has been installed and loaded. When needed, you can add DASD easily through the CP LINK command. This strategy helps you avoid additional steps to add new DASD.
- After you install Linux, check for Linux service packs and apply them before you clone Linux virtual servers.
- Ensure your Linux server /etc/resolv.conf points to the correct name server.
- If you want to use the signal shutdown function for all your Linux systems, you can make the required modification now so it will be included in all clones. See "Steps for enabling Linux virtual servers to shut down automatically" on page 87.
- If you want to monitor the performance of Linux virtual servers with the Performance Toolkit for VM, you might need to install additional software on the master Linux so all clones automatically get the monitoring software. See "Monitoring Linux virtual servers with Performance Toolkit for VM" on page 118.

Example of using FTP to get the Linux boot files

This topic is an example of how to use FTP to get the Linux boot files from a remote server.

Tip: If you do not have an external FTP server or do not want to create an external connection due to security concerns, you can install Linux by accessing a DVD or removable medium in the hardware management console (HMC) through FTP. For more information, see "Example of using FTP to install Linux from the hardware management console," on page 139.

In this example, parts in bold indicate commands you would issue. First, access as your A-disk a disk large enough to hold the boot files (the 192 disk in the example). Note the command `locsite fix 80`, which sets the VM file format to fixed length 80, the file format necessary for punching the binary files to the virtual machine reader.

Note: Because the directory containing the Linux boot files varies based on the Linux distribution and distribution level, the directory information used in the FTP GET commands in this topic is for example use only.

```
access 192 a
DMSACC724I 192 replaces A (191)
Ready;
ftp 9.185.246.25
VM TCP/IP FTP Level 54
Connecting to 9.185.246.25, port 21
220 linux390 FTP server (Version wu-2.4.2-VR17(1) Thu May 18 03:18:13 EDT 2000)
ready.
USER (identify yourself to the host):
user_ID
>>>USER user_ID
331 Password required for user_ID.
Password:
>>>PASS ********
230 User user_ID logged in.
Command:
bin
>>>TYPE i
200 Type set to I.
locsite fix 80
Command:
get /boot/s390x/vmrdr.ikr
>>>PORT 9,185,246,26,4,5
200 PORT command successful.
>>>RETR /boot/s390x/vmrdr.ikr
150 Opening BINARY mode data connection for vmrdr.ikr (1511884 bytes).
226 Transfer complete.
1511920 bytes transferred in 1.972 seconds. Transfer rate 766.69 Kbytes/sec.
Command:
get /boot/s390x/initrd suse.initrd
>>>PORT 9,185,246,26,4,7
200 PORT command successful.
>>>RETR /boot/s390x/initrd
150 Opening BINARY mode data connection for initrd (9862505 bytes).
8299680 bytes transferred.
226 Transfer complete.
9862560 bytes transferred in 12.366 seconds. Transfer rate 797.55 Kbytes/sec.
Command:
asc
>>>TYPE a
200 Type set to A.
Command:
get /boot/s390x/parmfile parm.file
>>>PORT 9,185,246,26,4,8
200 PORT command successful.
>>>RETR /boot/s390x/parmfile
150 Opening ASCII mode data connection for parmfile (38 bytes).
226 Transfer complete.
40 bytes transferred in 0.081 seconds. Transfer rate 0.49 Kbytes/sec.
Command:
quit
```

Example of punching Linux boot files to the virtual machine reader

This example shows commands used to punch Linux boot files to the virtual machine reader. When you punch files to the reader, note the spool identifiers (0126, 0127, and 0128 in the example–yours will differ) so if there are other reader files present you can queue (ORDER) the reader files in the proper order.

```
spool punch * close
Ready;
punch vmrdr ikr a (noh
RDR FILE 0126 SENT FROM LINMSTR PUN WAS 0126 RECS 3129 CPY  001 A NOHOLD NOKEEP
Ready;
punch parm file a (noh
RDR FILE 0127 SENT FROM LINMSTR PUN WAS 0127 RECS 3129 CPY  001 A NOHOLD NOKEEP
Ready;
punch suse initrd a (noh
RDR FILE 0128 SENT FROM LINMSTR PUN WAS 0128 RECS 3129 CPY  001 A NOHOLD NOKEE
Ready;
order reader 126 127 128
0000003 FILES ORDERED
Ready;
```

Tip: If you forget the spool file identifiers, issue QUERY RDR ALL.

Example of booting (IPL) the Linux boot files from the virtual machine reader

This example shows how you get the Linux installation and configuration started by booting (IPL) the virtual machine reader.

```
ipl 00c clear
.
.
.
Linux version release (root@ikr_rdr.suse.de)
.
.
.
Command line is: ramdisk_size=32768 root=/dev/ram0 ro
We are running under VM
This machine has no IEEE fpu
Initial ramdisk at: 0x02000000 (9862560 bytes)
Detected device 2000 on subchannel 0000 - PIM = 80, PAM = 80, POM = FF
Detected device 2001 on subchannel 0001 - PIM = 80, PAM = 80, POM = FF
Detected device 0202 on subchannel 0002 - PIM = C0, PAM = C0, POM = FF
Detected device 0192 on subchannel 0003 - PIM = C0, PAM = C0, POM = FF
Detected device 0200 on subchannel 0004 - PIM = C0, PAM = C0, POM = FF
Detected device 0201 on subchannel 0005 - PIM = C0, PAM = C0, POM = FF
Detected device 000C on subchannel 0006 - PIM = 80, PAM = 80, POM = FF
Detected device 000D on subchannel 0007 - PIM = 80, PAM = 80, POM = FF
Detected device 000E on subchannel 0008 - PIM = 80, PAM = 80, POM = FF
.
.
.
```

Here is an example of some of the errors you might see on your virtual machine console when formatting DASD during the Linux IPL. These errors are normal and are not a problem.

```
dasd_erp(3990):  /dev/dasda  ( 94:  0),0150@01: 074ab5c0: 00000000 00000000 0000
0000 00000000
dasd_erp(3990):  /dev/dasda  ( 94:  0),0150@01: 074ab5d0: 00000000 00000000 0000
0000 00000000
dasd_erp(3990):  /dev/dasda  ( 94:  0),0150@01: Failed CCW (074ab5b8) already lo
gged
end_request: I/O error, dev 5e:01 (dasd), sector 524320
EXT3-fs error (device dasd(94,1)): read_inode_bitmap: Cannot read inode bitmap -
 block_group = 2, inode_bitmap = 16777472
EXT3-fs error (device dasd(94,1)) in ext3_new_inode: IO failure
EXT3-fs error (device dasd(94,1)): ext3_new_inode: reserved inode or inode > ino
des count - block_group = 0,inode=1
EXT3-fs error (device dasd(94,1)) in ext3_new_inode: IO failure
EXT3-fs error (device dasd(94,1)): ext3_new_inode: reserved inode or inode > ino
des count - block_group = 0,inode=2
EXT3-fs error (device dasd(94,1)) in ext3_new_inode: IO failure
```

At this point, you would continue with instructions for your Linux distribution. See "Overview of defining virtual machines for Linux" on page 71.

Tip: When interacting with the Linux installation program, you might enter virtual machine console mode. For instance, the installation program might tell you to press Enter to accept a default option and pressing the Enter key results in the VM READ state. If this happens, simply press the Enter key again. For more information about virtual machine console states, see "The virtual machine console" on page 4. Also, see "Virtual machine operation tasks" on page 107.

(Optional) Steps for loading Linux automatically at logon

Once you are satisfied with your Linux master, you might want to automate the loading of Linux whenever its virtual machine is logged on. You can do this by placing an IPL command in the PROFILE EXEC.[4] Whenever the virtual machine is logged on, the PROFILE EXEC automatically executes and, through the IPL command, loads the Linux operating system.

Because the same PROFILE EXEC is replicated in every clone, this automation makes it convenient for systems with many Linux virtual servers. Chapter 9, "Setting up basic system automation," on page 85 shows you how to automate the startup of Linux virtual servers whenever z/VM itself loads; thus whenever z/VM loads, all your virtual machines are logged on and, in turn, each virtual machine loads the Linux operating system.

Before you begin: You need to be logged onto the LINMSTR virtual machine.

Perform these steps to load Linux automatically at logon:

1. If Linux is operating, shut it down.
2. Load (IPL) CMS. Issue this command:

```
ipl cms
z/VM V6.1.0    2004-02-11 16:57
.
.
.
Ready;
```

3. Edit the PROFILE EXEC. Issue this command:

```
xedit profile exec
```

4. At the bottom of the file, add this line:

```
.
.
.
"CP IPL 150 CLEAR"
```

5. Save the file. From the XEDIT command line, issue this command:

```
====> file
```

You are done.

4. As an alternative to using the PROFILE EXEC, you can use the COMMAND directory control statement to specify a command or a list of commands to be issued automatically whenever the Linux virtual machine is logged on. Note that the commands to be executed may be of any class, and that privileged commands can therefore be executed this way without having to give the virtual machine the associated privilege class. For more information, see the COMMAND directory control statement in *z/VM: CP Planning and Administration*, SC24-6178.

Chapter 8. Cloning Linux virtual servers

This topic covers a rudimentary way to replicate (or clone) Linux virtual servers. In Chapter 7, "Creating your first Linux virtual machine and installing Linux," on page 71, you created the LINUX prototype and installed the Linux operating system into LINMSTR. In this topic, you create a clone or replica of LINUX, copy the Linux file system from LINMSTR to the clone, and change the IP address of the clone in the Linux configuration files.

Steps for cloning a Linux virtual server

These steps tell you how to clone a Linux virtual server using DirMaint.

Before you begin: You need to

- Be sure that LINMSTR is not running so the new clone can use LINMSTR's IP address. You will use LINMSTR's IP address temporarily for each Linux clone during the cloning process. Since no two Linux virtual servers can have the same IP address at the same time, you must clone Linux servers one at a time, then customize them.
- Have the new IP address for the cloned Linux virtual server.
- Log on as MAINT.

Perform these steps to clone a Linux virtual server:

1. Add a clone of LINUX to the user directory. Issue this DirMaint command:

```
dirm add linux02 like linux pw new_password
```

   where you supply a password for *new_password*.

2. Unlock LINMSTR's directory entry. From the command line, type this command and press the Enter key:

```
dirm for linmstr unlock
DVHXMT1191I Your UNLOCK request has been sent for processing.
Ready;
 DVHREQ2288I Your UNLOCK request for LINMSTR at * has been accepted.
 DVHREQ2289I Your UNLOCK request for LINMSTR at * has completed; with
 DVHREQ2289I RC = 0.
```

3. Clone the disks from LINMSTR to the clone LINUX02. From the command line, type these DirMaint commands and press the Enter key after each one:

```
dirm for linux02 clonedisk 191 linmstr 191
dirm for linux02 clonedisk 150 linmstr 150
dirm for linux02 clonedisk 151 linmstr 151
```

   Tip: Cloning might take time. You are ready to move to the next step after your work units have completed successfully. To determine the status of your work units, type this command and press the Enter key:

```
dirm query datamove *
```

4. Log onto LINUX02 and IPL the Linux operating system.

5. Telnet to the new Linux virtual server and update the network configuration files to make permanent the network changes.

   Example: On a SUSE SLES9 system, you need to modify the IP address and host name through a tool like yast2.

6. Shut down the Linux operating system in LINUX02 and reboot. Check that the Linux operating system reboots properly and that the network connections are correct.

7. Try pinging this Linux server or try to access a remote site from this Linux server.

You know you are done when the Linux operating system reboots properly and the network connections are correct.

Chapter 9. Setting up basic system automation

This section covers basic forms of z/VM system automation:

- Starting and stopping virtual machines automatically
- Automating z/VM system and virtual machine operations through the programmable operator facility.

Starting and stopping virtual machines automatically

You can start and stop virtual machines manually, but it is more convenient to have z/VM do these tasks automatically. This topic shows you how to:

- Start your Linux virtual servers and important virtual machines automatically. You already configured DirMaint to start automatically in "Steps for automatically starting DIRMAINT" on page 59, and TCP/IP in "Steps for automatically starting TCP/IP" on page 63. Now you will do the same for other virtual machines.

  By including the CP XAUTOLOG command in the AUTOLOG1 virtual machine's PROFILE EXEC, a virtual machine starts automatically when z/VM is loaded (IPL). The AUTOLOG1 user ID is the default user that z/VM logs on at IPL time. When logged on, AUTOLOG1 executes its PROFILE EXEC, which can contain an XAUTOLOG command to start a virtual machine that runs Linux or any other vital system service. In turn, each virtual machine executes its own PROFILE EXEC, which contains the necessary commands to establish its operating environment; in the case of Linux, the PROFILE EXEC can load the Linux operating system (see "(Optional) Steps for loading Linux automatically at logon" on page 82.)
- Stop Linux virtual servers automatically. You can enable Linux to perform an orderly shutdown whenever you shut down z/VM or force the Linux virtual machine to log off with the FORCE command. Because Linux shuts down gracefully, Linux does not have to examine or repair disks when rebooting. Additionally, Linux can notify network monitors that it is shutting down.

  Prior to shutting down or forcing a virtual machine to log off, z/VM sends a signal to enabled virtual machines to shut down. The SYSTEM CONFIG file sets the amount of time that z/VM allows a virtual machine operating system to process a shutdown signal. In this topic, you configure Linux to process a shutdown signal.

Related information

z/VM: CP Planning and Administration, SC24-6178

Steps for automatically starting Linux virtual servers and other virtual machines

This procedure configures the AUTOLOG1 virtual machine to startup virtual machines automatically whenever z/VM starts.

Before you begin: You need to log on as MAINT.

Perform these steps to start virtual machines at startup (IPL) time:

1. Link to the AUTOLOG1 191 minidisk. Type this command and press the Enter key:

```
link autolog1 191 091 mr
Ready;
```

2. Access the 091 minidisk as Z. Type this command and press the Enter key:

```
access 091 z
Ready;
```

3. Edit AUTOLOG1's PROFILE EXEC.
   a. Type this command and press the Enter key:

```
xedit profile exec z
```

   b. Go to the bottom of the file. From the XEDIT command line, type this command and press the Enter key:

```
bot
```

   c. Move up one line (The last line logs off AUTOLOG1. You need to execute the XAUTOLOG commands before logging off AUTOLOG1). From the XEDIT command line, type this command and press the Enter key:

```
====> up 1
```

   d. From the XEDIT command line, type this command and press the Enter key:

```
====> input
```

4. Add these XAUTOLOG statements. Press the Enter key after you type each line:

```
ADDRESS COMMAND CP XAUTOLOG MONWRITE
ADDRESS COMMAND CP XAUTOLOG PERFSVM
```

   Note: These statements start the MONWRITE and PERFSVM virtual machines, which you are enabling in anticipation of setting up performance monitoring (see "Using the CP Monitor and Performance Toolkit for VM" on page 123).

5. For each Linux virtual server you want started automatically, add this statement:

```
ADDRESS COMMAND CP XAUTOLOG userid
```

   where *userid* is the user ID of the Linux virtual machine.

6. After every third XAUTOLOG statement that starts a Linux virtual server, add this statement:

```
ADDRESS COMMAND CP SLEEP 15 SEC
```

7. Save the file. Press the Enter key to leave input mode, then, at the XEDIT command line, type this command and press the Enter key:

```
====> file
```

8. Detach the 091 minidisk (AUTOLOG1's 191). Type this command and press the Enter key:

```
release z (det
```

9. If you want to test the autolog procedure, from the command line type this command and press the Enter key:

```
xautolog autolog1
```

You are done when the autolog procedure succeeds.

Steps for enabling Linux virtual servers to shut down automatically

Before you begin: You need to have a Linux kernel that supports a shutdown signal, such as SUSE SLES9. Check your Linux system documentation to determine if it supports the shutdown signal.

You need to log onto MAINT. You also need to log onto your Linux system as a superuser.

Perform these steps to enable Linux virtual servers to shut down automatically:

1. Log on as MAINT on z/VM and check whether your Linux virtual servers are enabled for the shutdown signal. Type this command and press the Enter key.

   Note: Your Linux operating systems need to be running to see if they are enabled for the shutdown signal.

```
q signals
```

Example:

```
q signals
                                        Signalled  Timeout
Userid        Signal    Signal Status   By         Remaining
LINUX01       SHUTDOWN  Enabled         -          -
LINUX02       SHUTDOWN  Enabled         -          -
LINUX03       SHUTDOWN  Enabled         -          -
LINUX04       SHUTDOWN  Enabled         -          -
LINUX05       SHUTDOWN  Enabled         -          -
Ready;
```

2. While logged on as a superuser on Linux, check the etc/inittab file for these lines:

```
 # z/VM or LPAR is shutting down
ca:12345:ctrlaltdel:/sbin/shutdown -h now
```

Important: Be sure the shutdown command shuts down Linux (-h) rather than restarting it (-r).

You are done.

Setting up the programmable operator

This topic describes the programmable operator facility and shows you how to set it up.

Overview of the programmable operator

The programmable operator facility increases the efficiency of z/VM system operation and allows operation of systems in a distributed processing environment. The facility intercepts all messages and requests directed to the z/VM operator virtual machine and handles them according to programmed actions. Some messages are simply recorded for future reference, others are acted upon programmatically, and others are sent to a real operator to handle.

The programmable operator facility runs in a CMS virtual machine. Although it can run in any virtual machine, because of its programmed capability to log, handle, or redirect messages, it is most commonly run in the CP system operator's virtual machine.

The programmable operator facility compares all messages directed to it against entries listed in a *routing table*, a CMS file. When a match occurs, the prescribed action is performed. If there is no match, no action is performed. Any messages requiring a real operator's response or action are sent to the defined operator at another console, called the *logical operator* console. If the default action is to log messages, all messages are logged in a CMS *log file*. The benefit is that the real system operator sees only important messages, while all messages are recorded for auditing.

Example: As shown in Figure 8 on page 89, the OPERATOR user ID is defined as the CP system operator when the CP system is configured. Set up the programmable operator to run in the OPERATOR virtual machine and establish another virtual machine with a user ID of LGLOPR. In the routing table, specify LGLOPR as the logical operator. Now any CP or virtual machine user messages sent to the system operator virtual machine can be handled or filtered by the programmable operator or routed to the LGLOPR user ID. A real system operator logs onto LGLOPR and monitors messages sent to that virtual machine.

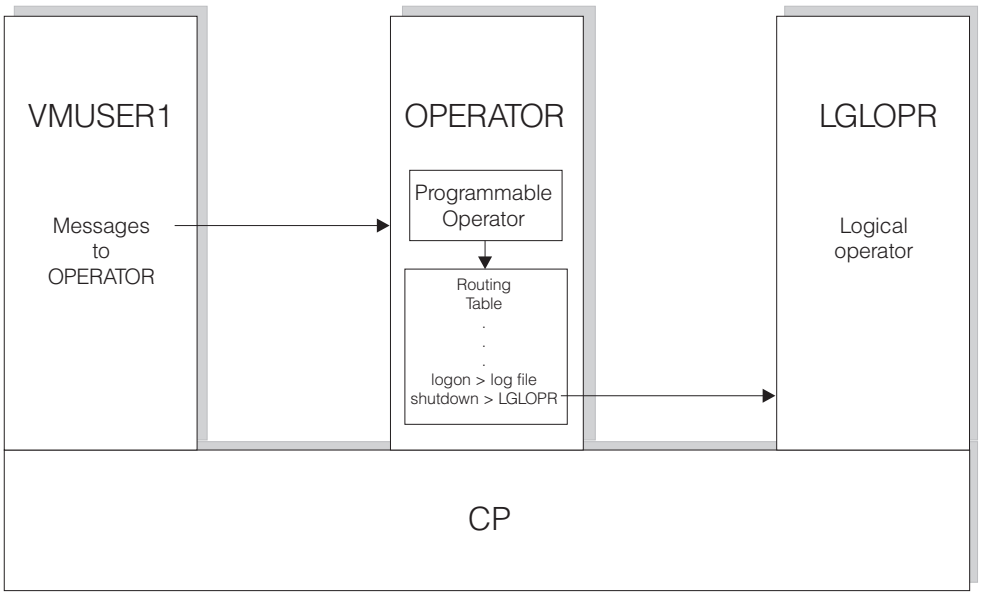

Figure 8. Example of a programmable operator

Related information

"The Programmable Operator Facility" in *z/VM: CMS Planning and Administration*, SC24-6171

Steps for setting up the routing table

This procedure has you modify the PROP RTABLE and add comparison entries for your Linux virtual servers. A comparison entry is a line in the PROP RTABLE that matches a message sent to the programmable operator to an action, such as logging the message or sending the message to the logical operator. For instance, one comparison entry is designed to send a message to the logical operator if a Linux system abnormally terminates. For more information about the statements in the PROP RTABLE, including comparison entries, see "The Routing Table" in *z/VM: CMS Planning and Administration*, SC24-6171.

Use this procedure as a model by which to automate other actions for your Linux virtual servers.

Before you begin: If OPERATOR is not logged on, you need to log onto the OPERATOR virtual machine. OPERATOR needs to be running CMS (see "Step for modifying the OPERATOR's directory entry" on page 60).

Rules: When editing the routing table, you must:

- Keep entries in their proper columns.
- Right-justify numbers in their columns.

Perform these steps to set up the routing table:

1. Copy the sample routing table (PROP RTABLE) file from the CMS 190 minidisk. Because the file is mode 5 (hidden), you must access the 190 disk as C/A to copy the file. Type these commands and press the Enter key after each one:

```
access 190 c/a
DMSACP723I C (190) R/O
DMSACP725I 190 also = S disk
Ready;
copyfile prop rtable c = = a1
Ready;
rel c
Ready;
```

2. Edit PROP RTABLE A1. Type this command and press the Enter key:

```
xedit prop rtable a
```

   a. To assure that your typing is always uppercase, from the XEDIT command line, type this command and press the Enter key:

```
====> set case u
```

   b. Set line numbers on. From the XEDIT command line, type this command and press the Enter key:

```
====> set num on
```

3. Locate the LGLOPR statement.
   a. Replace "OPERATOR" with "LGLOPR".
   b. Remove the string "HOSTNODE".

   Result:

```
00001 *        ----- SPECIFY THE PROP CONFIGURATION -----
00002
00003 * IDENTIFY THE LOGICAL OPERATOR
00004
00005 LGLOPR LGLOPR
```

4. Delete these lines (26 to 31). Use the block delete ("dd") prefix commands in the prefix area:

```
dd    /LOGON                    21  26  3
00027 /LOGOFF$¬FORCED           21  80  3
00028 /DISCONNECT               21  31  3
00029 /RECONNECT                21  30  3
00030 /DIAL                     21  25  3
dd    /DROP                     21  25  3
```

5. Delete all lines with "PROPNODE" and "*NCCF" in them. From the XEDIT command line, type these commands and press the Enter key after each one:

```
====> all/propnode
```

```
====> delete *
```

```
====> all/NCCF
```

```
====> delete *
```

```
====> all
```

6. Save the file. From the XEDIT command line, type this command and press the Enter key:

```
====> save
```

7. Move the current line to the line **before** "SEND ALL OTHER TRAPPED DATA TO THE LOGICAL OPERATOR". From the XEDIT command line, type these commands and press the Enter key after each one:

```
====> bot
```

```
====> u3
```

```
====> set case mixed
```

"SET CASE MIXED" instructs XEDIT to keep characters you enter in mixed case.

```
====> input
```

8. Add these lines. Remember to keep entries in their proper columns and keep characters in mixed case as shown.

Note: These lines filter many messages. Over time, you might want to remove some if you find you do not use them or want to handle certain messages in other ways, such as logging them.

```
*----------------------- --- --- -- -------- -------- -------- --------
* NOTIFY LOGICAL OPERATOR IF LINUX ABENDS
*----------------------- --- --- -- -------- -------- -------- --------
/OOPS/                            8                  DMSPOS   LGLOPR
$DISABLED                         8                  DMSPOS   LGLOPR
$PSW$                             8                  DMSPOS   LGLOPR
$psw$                             8                  DMSPOS   LGLOPR
$disconnected/                    8                  DMSPOS   LGLOPR
/HCP$                             8                  DMSPOS   LGLOPR
$FAILURE/                         8                  DMSPOS   LGLOPR
$failed/                          8                  DMSPOS   LGLOPR
$Failed/                          8                  DMSPOS   LGLOPR
$No such/                         8                  DMSPOS   LGLOPR
$ERROR/                           8                  DMSPOS   LGLOPR
$error/                           8                  DMSPOS   LGLOPR
$_MACHINE$                        8                  DMSPOS   LGLOPR
$cannot open/                     8                  DMSPOS   LGLOPR
*----------------------- --- --- -- -------- -------- -------- --------
* DON'T WORRY ABOUT ANY OTHER SCIF OUTPUT FROM MONITORED USERS
*----------------------- --- --- -- -------- -------- -------- --------
                                  8
*----------------------- --- --- -- -------- -------- -------- --------
```

9. Save the PROP RTABLE file. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

You are done.

Steps for setting up the programmable operator

Because z/VM automatically logs on OPERATOR, you can have the programmable operator automatically invoked without the need to start it manually whenever z/VM initializes. In these steps, you set up the programmable operator to start automatically in the OPERATOR virtual machine.

Before you begin: If OPERATOR is not logged on, you need to log onto the OPERATOR virtual machine. OPERATOR needs to be running CMS (see "Step for modifying the OPERATOR's directory entry" on page 60).

Perform these steps to set up the programmable operator:

1. Log onto OPERATOR and edit its PROFILE EXEC. Type this command and press the Enter key:

```
xedit profile exec
```

2. Go to the bottom of the file and enter input mode. From the XEDIT command line, type these commands and press the Enter key after each one:

```
====> bot
```

```
====> input
```

3. Add a line with this statement:

```
EXEC PROPST PROP DISCONN
```

4. Press the Enter key twice, then save the file. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

5. Check that the programmable operator works.
   a. Log onto LGLOPR.
   b. From **OPERATOR**, reload CMS. From the command line, type this command and press the Enter key:

```
ipl cms
```

      Result: CMS initializes, executes the PROFILE EXEC, then the OPERATOR virtual machine disconnects. You see initialization statements on LGLOPR's virtual console.

You are done when you see OPERATOR disconnect and the programmable operator initialized.

Important: Once the programmable operator is running, do not log onto OPERATOR except when necessary. If you do log on to OPERATOR, before you can issue commands other than those that control the programmable operator, you must issue the STOP command.

Steps for automating Linux virtual consoles

This task shows you how to send Linux virtual console[5] messages to the programmable operator. Because your Linux virtual servers typically run disconnected, you do not see virtual console messages and thus do not know if problems are occurring in the Linux virtual machines. Through this task, you send virtual console messages to the programmable operator for handling. By modifying the routing table, you can record incoming messages or forward important messages to the real operator. Figure 9 shows how automating Linux virtual consoles works.

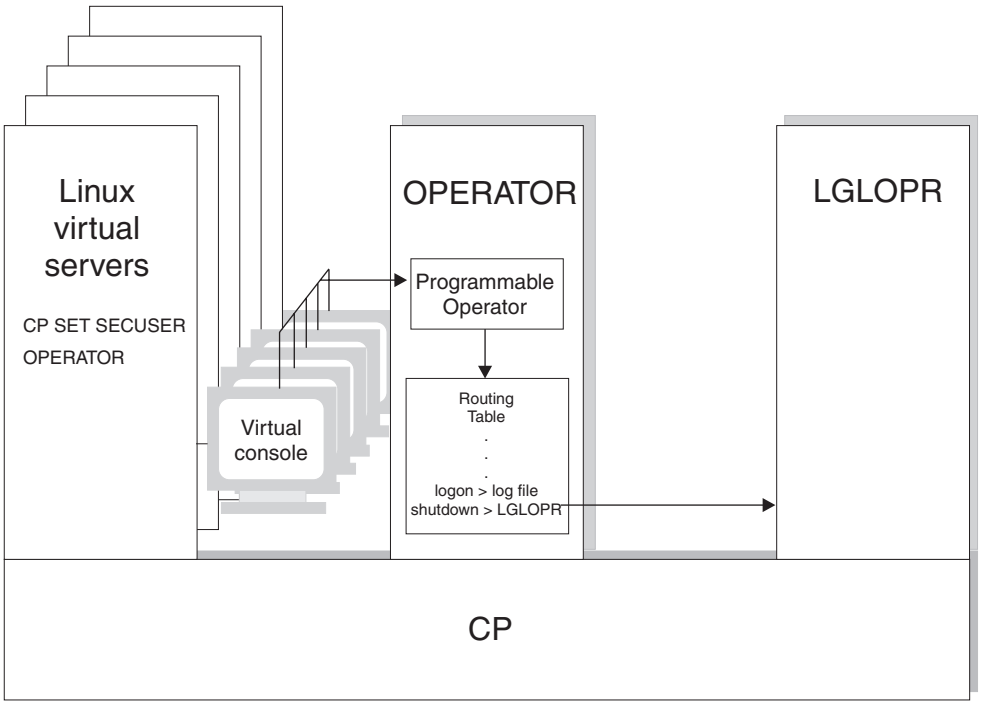

Figure 9. Automating Linux virtual consoles

In the figure, each Linux virtual machine has the command CP SET SECUSER OPERATOR in its PROFILE EXEC. The command defines the OPERATOR user ID as a *secondary console*, meaning all virtual console messages are sent to OPERATOR. Because OPERATOR is running the programmable operator facility, you can filter messages coming from the Linux virtual machines. You can log the insignificant messages for later reference and forward messages requiring special handling to a real operator at LGLOPR.

By using the secondary console, you can also send messages to the Linux virtual servers to do certain actions, such as attach a network interface (COUPLE command), should a Linux failure be detected.

5. For a description of virtual consoles, see "Overview of console types" on page 97.

Before you begin: You need to set up the programmable operator. See "Steps for setting up the programmable operator" on page 92. You need to be able to log onto your Linux virtual machines, but Linux must be shut down.

Perform these steps to automate Linux virtual consoles:

1. Log onto a Linux virtual machine.
2. If Linux is loaded automatically or is already running:
   a. Shut Linux down.
   b. Load (IPL) CMS with the ACC (NOPROF option.
      1) From the command line, type this command and press the Enter key:

```
ipl cms
```

      2) When you see the CMS initialization message and VM READ, type this command and press the Enter key:

```
i cms
z/VM V6.1.0    2004-07-27 14:44









acc (noprof
                                                        VM READ   VM610A
```

3. Edit the PROFILE EXEC. From the command line, type this command and press the Enter key:

```
xedit profile exec a
```

4. Go to the bottom of the file. From the XEDIT command line, type this command and press the Enter key:

```
====> bot
```

5. In the prefix area **before** the "CP IPL 150 CLEAR" statement, type "i" and press the Enter key:

```
i     "ACCESS 592 Z"          /* Get to TCP/IP functions  */
00009 "CP IPL 150 CLEAR"
```

6. Type this line:

```
"CP SET SECUSER OPERATOR" /* Operator is sec console*/
```

Result: The PROFILE EXEC looks like this:

```
/*  Sample PROFILE EXEC for Linux servers */
"CP TERM LINEND %"      /* Use %CP to talk to CP    */
"CP SET PF12 RETRIEVE"  /* Recall previous commands */
"CP TERM MORE 1 0"      /* Clear screen after 1 sec */
"CP SET RUN ON"         /* CP READ won't stop server*/
"CP TERM HOLD OFF"      /* (Look in book)           */
"ACCESS 592 Z"          /* Get to TCP/IP functions  */
"CP SET SECUSER OPERATOR" /* Operator is sec console*/
"CP IPL 150 CLEAR"      /* Load Linux automatically */
```

7. Save the PROFILE EXEC. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

8. From the command line, type these commands and press the Enter key after each one:

```
profile
%cp disconnect
```

9. Repeat steps 1 to 8 for each Linux virtual machine.

Continue to the next procedure.

Steps for testing your automation

To test your automation, try to attempt to couple a network adapter that is already coupled. Messages at the LGLOPR virtual console that the coupling fails tell you that your automation is working properly.

Before you begin: You need to be logged onto LGLOPR and you must have at least one Linux virtual server running.

Perform these steps to test your automation:

1. Reset the terminal line end character for LGLOPR. From the command line, type this command and press the Enter key:

```
term linend #
Ready;
```

2. From the command line, type this command and press the Enter key:

```
#cp send userid %cp couple 600 system vswitch1
```

Where *userid* is the user ID of a Linux virtual server.

Result: You should see these failure messages on the LGLOPR virtual console, which proves your automation is working:

```
Ready;
 userid   systemid: HCPCPL2788E NIC 0600 not connected; already connected to VS
 WITCH SYSTEM
 userid   systemid:    ..  VSWITCH1
```

Where *userid* is the user ID of the Linux virtual server and *systemid* is the z/VM system ID.

You know you are done when the test succeeds.

Chapter 10. Performing run-time tasks

This topic covers proactive tasks you need to do to keep z/VM and Linux virtual servers running efficiently. It includes ways for you to monitor z/VM operations so you can prevent system problems.

For monitoring system performance, see Chapter 11, "Monitoring performance and capacity," on page 117.

Overview of console types

During run-time operations, you can use several consoles. Consoles can be located on the same physical device (workstation, PC, display) or on separate devices. Because of the flexibility of z/VM, the number of consoles varies: you might have a few or many Linux production systems, each with their own console. Regardless of the number of consoles, these are the types of consoles available and a description of each:

- Hardware management console. The hardware management console communicates with the hardware. From this console you load (IPL) z/VM itself. You can also do basic z/VM system management from the hardware management console. For details, consult the documentation for your hardware management console.
- Integrated ASCII console. The integrated ASCII console can be used to communicate with a Linux guest without the need for any special connectivity or additional hardware, such as control units or network connections. z/VM itself does not exploit the ASCII console, but z/VM can attach it to and detach it from a Linux guest, and query the status of the console. z/VM manages the console as a dedicated device; that is, only one Linux guest can use it at a time. Your Linux guest must be configured to use the ASCII console.

  The console provides a VT220 interface for Linux, so you can use Linux tools such as vi and emacs. In an emergency, when no other connectivity is available, this connection could be used, though the console is not limited to such use. For more information, see "Step for managing real devices" on page 100.
- System operator console. The system operator console is the console at which CP automatically logs on the primary system operator virtual machine at z/VM load time. In "Steps for setting addresses for consoles" on page 46, you set the real address for this console.

  If you do not use the programmable operator facility, you perform system tasks from this console, such as loading z/VM, shutting z/VM down, controlling real devices, communicating with users, and responding to z/VM problems.
- Logical operator console. If you followed "Setting up the programmable operator" on page 88, the system console is automated, and you perform system tasks from the logical operator console. In this section, it is assumed you perform real system tasks from the logical operator console.
- Virtual console for a virtual machine. When you log onto z/VM as a user, the session you enter is the virtual console. You use this console to control Linux's virtual machine. The tasks you can do from the virtual console include loading Linux, defining the storage size of the virtual machine, and defining virtual CPUs.
- Linux's system console. The virtual console becomes the Linux system console once Linux is loaded. Linux system messages are displayed on this console.

"Setting up the programmable operator" on page 88 shows you how to send Linux console messages to one central operator.

This section divides up run-time tasks into tasks you can do from the logical operator console (real operation tasks) and tasks you can do from the virtual console (virtual operation tasks). Because the Linux virtual machines have class G command authority and some tasks require the authority of the primary system operator, you must use the logical operator console for those tasks. Other tasks (those for a particular virtual machine running Linux) are executed from the virtual console.

On the logical operator console, you receive messages from the z/VM operating system and from z/VM users (virtual machines). The operating system sends you messages about hardware errors, software errors, and resource shortages. z/VM users who need to have tapes or DASD volumes mounted might send messages to make their requests. Watching the logical operator console lets you respond quickly to system errors and z/VM user requests.

Some tasks you perform from the Linux virtual console are virtual machine tasks; those are documented in this section. Other tasks are operations you perform for the Linux operating system; for those tasks, you must follow the documentation for your Linux distribution.

Real operation tasks

Step for monitoring the logical operator console

You will spend much of your time watching the z/VM system for changes in system availability and for requests from users for resources.

Before you begin: You must be able to log on as MAINT or onto service virtual machines as indicated.

Perform this step to watch the z/VM logical operator console:

- Base your actions on this table:

  Tip: If you need help understanding the message or knowing what to do, use the Help facility. From the command line, type this command and press the Enter key:

  ```
  help msg message_id
  ```

  where *message_id* is the message identifier. **Example:**

  ```
  help msg hcplnm053e
  ```

If you receive ...	Then ...
LOGON or LOGOFF messages	Depending on the message, you might need to take action. See "Step for managing users" on page 105.

If you receive ...	Then ...
A paging capacity warning: `90 percent of all paging space is in use.`	• Review system usage and take steps to reduce the system load. • Log on as MAINT and add new paging volumes: 1. Format and allocate a new paging volume. See "Steps for adding a paging, spooling, or user volume" on page 35. 2. Attach the new paging volume to the system: `attach rdev to system` where *rdev* is the real device address. 3. Update the SYSTEM CONFIG file to include the new paging volume. See "Steps for updating the CP-owned volume list" on page 37.
A spooling capacity warning: `90 percent of all spooling space is in use.`	• Review the spool files. You can start additional printers to reduce print spool backlog. • If a reader backlog exists, request all users to read in their files. • Purge unneeded spool files. You can use SPTAPE to save selected spool files on tape before purging them. • Log on as MAINT and add new spooling volumes: 1. Format and allocate a new spooling volume. See "Steps for adding a paging, spooling, or user volume" on page 35. 2. Attach the new spooling volume to the system: `attach rdev to system` where *rdev* is the real device address. 3. Update the SYSTEM CONFIG file to include the new spooling volume. See "Steps for updating the CP-owned volume list" on page 37.
Device errors	See "Step for managing real devices" on page 100.

Warnings about journaling, erep, and accounting recording:
`{EREP | Accounting | SYMPTOM | CONFIG} records are accumulating for userid userid.`
	1. Log onto the user ID indicated in the message. When CMS begins to boot and you see VM READ, type this command and press the Enter key:
	`access (noprof`
	2. Archive the data on the A-dsik according to your retention policy.
	3. Erase the data files on the A-disk.
	4. Log off.
	5. From LGLOPR, type this command and press the Enter key:
	`xautolog userid`

Complaints about system performance	See Chapter 11, "Monitoring performance and capacity," on page 117.

You know you are done when the system and users are running efficiently.

Step for restarting z/VM

If you need to restart z/VM, use the SHUTDOWN REIPL command. The SHUTDOWN command sends signals to the Linux virtual servers to shut down and waits for a period of time defined in the SYSTEM CONFIG file.

Before you begin: You must log on as the system operator (if you followed "Setting up the programmable operator" on page 88, that user is LGLOPR).

Perform this step to restart z/VM:

- Type this command and press the Enter key:

```
shutdown reipl
```

  Result: After the system shuts down and reboots (IPLs), you see a number of IPL messages. z/VM restores the system to the same state as it was prior to shutdown.

You know you are done when you see z/VM restored.

Step for managing real devices

Before you begin: You must log on as the system operator (if you followed "Setting up the programmable operator" on page 88, that user is LGLOPR).

Perform this step to manage devices:

- Base your actions on this table:

  Tip: If you need more information about a command, use the Help facility. From the command line, type this command and press the Enter key:

```
help cp command_name
```

  where *command_name* is the CP command.

  Example:

```
help cp vary
```

If you want to ...	Then use this command ...

Use the ASCII console to communicate with a Linux virtual machine	CP ATTACH SYSASCII TO *userid*
	
	Notes:
	1. Refer to the Red Hat, Inc., and SUSE Linux configuration tools for more information about the ASCII console support. Also, see the console support section in *Linux on System z: Device Drivers, Features, and Commands* on the IBM developerWorks Linux on System z Web site entitled "Documentation for Development stream" at:
	http://www.ibm.com/developerworks/linux/linux390/documentation_dev.html
	2. For Linux to use the ASCII console, the Linux kernel must enable the vt220 sclp and `/dev/ttyS1` must be uncommented in the inittab. For SUSE Linux and Red Hat, Inc., the vt220 console is enabled by default, but you must uncomment `/dev/ttyS1` in the inittab.
	
	Example:
	`attach sysascii to linux1`
	`SYSASCII attached to LINUX1`
	`Ready;`

If you want to ...	Then use this command ...

Display the status of real devices	CP QUERY *rdev1 rdev2 ...*
	
	Examples:
	• For DASD:
	`query 280 7806`
	`DASD 0280 CP SYSTEM XAUSR1   49`
	`DASD 7806 ATTACHED TO ALAN 7806 R/W 610SPL`
	`Ready;`
	
	In the first response line, the first number listed is the real device number of the DASD. "SYSTEM" indicates the device is allocated to the system for use as users' minidisks. "XAUSR1" is the volume label of the device. "49" indicates there are 49 active links to the volume.
	
	In the second response line, "ATTACHED TO ALAN" indicates the DASD is dedicated to ALAN's virtual machine for its exclusive use.
	`query db41`
	`14:38:46 DASD DB41 CP OWNED 610RES 79`
	`Ready;`
	
	If you see "OWNED", the device is used by the system for paging or spooling activity.
	• For tape drives:
	`query 102`
	`TAPE 0102 ATTACHED TO RALPH 0236 R/W`
	`Ready;`
	
	This means that the magnetic tape drive at the address 0102 is attached in read/write mode to RALPH's virtual machine as the virtual device number 0236.
	• For displays:
	`query 0bcd`
	`GRAF  0BCD ATTACHED TO OLIVER 0010`
	`Ready;`
	
	This message means the display device at the real device number 0BCD is attached to OLIVER's virtual machine as the virtual device 0010.
	• For Open Systems Adapter devices:
	`query 480`
	`OSA 0480 ATTACHED TO LINUX1  0480`
	`Ready;`
	
	The first number is the real device number of the Open Systems Adapter device. The last number is the virtual device number which each virtual machine uses to refer to an OSA device.

If you want to ...	Then use this command ...

Display the status of the cryptographic facility	CP QUERY CRYPTO APQS
	
	Example:
	`q crypto apqs`
	`AP 02 CEX2C Queue 08 is reserved for dedicated use`
	`AP 02 CEX2C Queue 09 is superseded by CEX2A`
	`AP 02 CEX2C Queue 10 is superseded by CEX2A`
	`AP 02 CEX2C Queue 11 is dedicated to GARDNERK`
	`AP 03 CEX2C Queue 08 is superseded by CEX2A`
	`AP 03 CEX2C Queue 09 is superseded by CEX2A`
	`AP 03 CEX2C Queue 10 is superseded by CEX2A`
	`AP 03 CEX2C Queue 11 is dedicated to GARDNERK`
	`AP 08 CEX2A Queue 08 is reserved for dedicated use`
	`AP 08 CEX2A Queue 09 is installed`
	`AP 08 CEX2A Queue 10 is installed`
	`AP 08 CEX2A Queue 11 is installed`
	`AP 09 CEX2A Queue 08 is installed`
	`AP 09 CEX2A Queue 09 is installed`
	`AP 09 CEX2A Queue 10 is installed`
	`AP 09 CEX2A Queue 11 is installed`
	`Ready;`

Enable a logical connection to a real device	CP VARY ONLINE *rdev*
	
	Example:
	`vary online 255`
	`0255 VARIED ONLINE`
	`Ready;`

Disable a logical connection to a real device	CP VARY OFFLINE *rdev*
	
	Example:
	`vary offline 255`
	`0255 VARIED OFFLINE`
	`Ready;`

Find out channel path status by device	CP QUERY PATHS TO *rdev*
	
	Example:
	`query paths to 0291`
	`Device 0291, Status ONLINE`
	`CHPIDs to Device 0291 (PIM) : 19 1B 2E 2F`
	`Physically Available (PAM) : + + + +`
	`Online (LPM) : + + + +`
	`Legend + Yes - No`
	`Ready;`
	
	In response, CP lists all the channel paths that are physically available to the device and indicates whether the channel paths are logically online or offline.

If you want to ...	Then use this command ...

Enable a logical path between the host system and one or more real devices.	CP VARY ON PATH *nn* TO *rdev*
	
	Example: To make path C2 leading to device 140 available to that device:
	
	`vary on path c2 to 140`
	`Vary path c2 online command initiated`
	`Vary path c2 online command complete`
	`Ready;`

Disable a logical path between the host system and one or more real devices.	CP VARY OFF PATH *nn* TO *rdev*
	
	Example: To make path C2 unavailable to devices 0190 through 0197:
	
	`vary off path c2 from 0190-0197`
	`Vary path c2 offline command initiated`
	`Path c2 not varied offline from device 0192`
	`because it is the last path to device`
	`Vary path c2 offline command complete`
	`Ready;`

Display the logical path status and devices with that logical path installed	CP QUERY CHPID *nn*
	
	Example: To find out the status of channel path 1A:
	
	`query chpid 1a`
	`PATH 1A ONLINE TO DEVICES 0190, 0191, 0193`
	`PATH 1A OFFLINE TO DEVICES 0192`
	`Ready;`

Make a channel path available to a device	CP VARY ONLINE CHPID *nn*
	
	Example:
	
	`vary online chpid 21`
	`Channel path 21 was successfully varied online`
	`Ready;`

Remove a channel path from a device	CP VARY OFFLINE PATH *nn* FROM *rdev*
	
	Example:
	
	`vary offline chpid 21`
	`Channel path 21 was successfully varied offline`
	`Ready;`

Connect a real device (for example DASD or OSA-Express) to the virtual machine	ATTACH *rdev userid*
	
	Example:
	
	`att 2306 lantzy`
	`OSA 2306 ATTACHED TO LANTZY 2306`
	`Ready;`

If you want to ...	Then use this command ...

Add expanded storage to the virtual machine	ATTACH XSTORE *userid*
	
	Issue this command when Linux is shutdown.
	
	Example:
	`attach xstore lantzy`
	`15:49:56 XSTORE MIGRATION STARTED, MAY TAKE SEVERAL MINUTES TO COMPLETE`
	`15:49:56 XSTORE CLEARING STARTED`
	`15:49:57 XSTORE attached, size= 512M`
	`Ready;`

You are done when you have successfully managed the real devices.

Step for managing users

Before you begin: You need to log on as the system operator (if you followed "Setting up the programmable operator" on page 88, that user is LGLOPR) or MAINT, as indicated.

Perform this step to manage users:

- Base your actions on this table:

  Tip: If you need more information about a command, use the Help facility. From the command line, type this command and press the Enter key:

  ```
  help cp command_name
  ```

  where *command_name* is the CP command.

  Example:

  ```
  help cp force
  ```

If you want to ...	Then use this command ...

Add a minidisk to a Linux virtual server	DIRM FOR *userid* AMDISK *vdev* 3390 AUTOV *cyls volid*
	
	Issue this command from MAINT.
	
	Example:
	`dirm for linmstr amdisk 291 3390 autov 5 LINUX3`
	`DVHXMT1191I Your AMDISK request has been sent for processing.`
	`Ready;`
	` DVHREQ2288I Your AMDISK request for LINMSTR at * has been accepted.`
	` DVHSCU3541I Work unit 04112907 has been built and queued for processing.`
	` DVHSHN3541I Processing work unit 04112907 as MAINT from ZVMLINUX,`
	` DVHSHN3541I notifying MAINT at ZVMLINUX, request 33 for LINMSTR sysaffin`
	` DVHSHN3541I *; to: AMDISK 0291 3390 AUTOV 5 LINUX3`
	` DVHBIU3450I The source for directory entry LINMSTR has been updated.`
	` DVHDRC3428I Changes made to directory entry LINMSTR have just been`
	` DVHDRC3428I placed online.`
	` DVHSHN3430I AMDISK operation for LINMSTR address 0291 has finished (WUCF`
	` DVHSHN3430I 04112907).`
	` DVHREQ2289I Your AMDISK request for LINMSTR at * has completed; with`
	` DVHREQ2289I RC = 0.`

If you want to ...	Then use this command ...

Display the number of logged-on users	CP QUERY USERS
	
	Issue this command from LGLOPR.
	
	Example:
	
	`query users`
	`8 USERS,  0 DIALED,  0 NET`
	`Ready;`

Display a user ID and device address of the user	CP QUERY *userid*
	
	Issue this command from LGLOPR.
	
	Example:
	
	`query linux1`
	`LINUX1  -L008B`
	`Ready;`

List logged on users and addresses	CP QUERY NAMES
	
	Issue this command from LGLOPR.
	
	Example:
	
	`query names`
	`LINUX1  -L008E, LINUX2  -L0094, LINUX3  -L008B, MONWRITE - DSC`
	`LINUX4  -L0082, LINUX5  -L003E, LINUX6  -L008A, LINUX6  -L0086`
	`Ready;`
	
	"DSC" means the virtual machine is running in disconnected mode.

Log off an active user	CP FORCE *userid*
	
	Issue this command from LGLOPR.
	
	Example:
	
	`FORCE LINUX01`
	`GRAF  0020 LOGOFF AS  LINUX01  USERS = 875  FORCED BY LGLOPR`

Log on another virtual machine	CP XAUTOLOG
	
	Issue this command from LGLOPR.
	
	Example:
	
	`xautolog linux03`
	`Command accepted`
	`AUTO LOGON *** LINUX03  USERS = 24 BY LGLOPR`
	`Ready;`

Permanently change the line end character for a user (the user's virtual console)	DIRM FOR *userid* LINEND *char* Issue this command from MAINT. **Example:** `dirm for linux01 linend pct`

If you want to ...	Then use this command ...
Permanently change the storage size of a virtual machine	DIRM FOR *userid* STORAGE *nnn*M Issue this command from MAINT. **Example:** `dirm for linux01 storage 256M`

Check to see whether a Linux virtual server is enabled for the shutdown signal	QUERY SIGNAL
	
	Issue this command from LGLOPR.
	
	Example:
	`query signal`
	`Userid Signal Signal Status Signalled By Timeout Remaining`
	`LINUX01 SHUTDOWN Enabled - -`
	`LINUX02 SHUTDOWN Enabled - -`
	`LINUX03 SHUTDOWN Enabled - -`
	`LINUX04 SHUTDOWN Enabled - -`

Signal a virtual machine to shut down	CP SIGNAL SHUTDOWN *userid* WITHIN *seconds*
	
	Issue this command from LGLOPR.
	
	Example:
	`signal shutdown linux04 within 300`
	`Ready;`

You are done when you have successfully managed the users.

Virtual machine operation tasks

When you run Linux under z/VM, there are times when you must simulate real hardware functions. Use these CP commands to simulate these real operator functions. Issue CP commands from the Linux virtual console, which is the 3270 session for the virtual machine you log onto.

Tips:

- To use any CP commands, you must precede them with the line end character plus "cp"; if you followed "Steps for updating special escape character defaults" on page 47, that string is "%cp". As shipped, z/VM uses "#cp".
- If you are on the virtual console and need to break out of a Linux command, Linux requires a key combination of the logical Not (X'5F') plus "c", which is equivalent to CNTRL-C. The logical Not symbol is a nonstandard character. Most code pages assign the logical Not symbol to the circumflex key (^). Check your code page for the key that corresponds to the logical Not symbol.

Steps for using CP commands at the Linux virtual console

Before you begin: You must have authority to log on the virtual machine that runs Linux.

Perform these steps to use CP commands at the Linux virtual console:

1. Log on the virtual machine that runs Linux.
2. Base your action on the choices in the table:

If you want to ...	Then use this command ...
Change the line end character	TERMINAL LINEND *new_char* **Example:** `%cp terminal linend +` **Tips:** • If you followed "Steps for updating special escape character defaults" on page 47, you defined "%" as a system-wide default line end character. Use this command if you need to change the line end character for the virtual console (3270 session) for an individual Linux server. Note that after changing the line end character, you must use the newly-defined line end character instead of the default. • Some Linux distributions support the hcp utility, which allows you to issue CP commands from the virtual console or telnet session by prefacing the command with "hcp" rather than the line end character plus "cp". Newer Linux distributions support the vmcp module/command, which allows you to substitute "vmcp" rather than the line end character plus "cp". Consult documentation for your distribution to determine whether it supports the hcp utility or vmcp.

Disconnect your display from z/VM without stopping operations in your virtual machine	DISCONNECT
	
	Examples:
	`%cp disconnect`
	`DISCONNECT AT hh:mm:ss zone weekday mm/dd/yy`
	
	If you use "vmcp" to address CP:
	`vmcp disconnect`
	`DISCONNECT AT hh:mm:ss zone weekday mm/dd/yy`
	
	Tips:
	• Once you issue DISCONNECT, you remain disconnected until you enter the LOGON command.
	• When your virtual machine is reconnected using the usual logon procedure, it might be placed in CP console function mode (CP READ). If so, to resume execution on your virtual machine, enter the BEGIN command.

If you want to ...	Then use this command ...

Change the memory size of the virtual machine	DEFINE STORAGE
	
	Attention:
	• Issue this command only when Linux is shut down because changing storage sizes resets the virtual machine and any operating system running in it.
	• The maximum size of your virtual machine is defined in your directory entry. If you need more storage than the maximum, contact your system administrator to increase the maximum size.
	
	Examples:
	`%cp define storage 512m`
	`STORAGE = 512M`
	`Storage cleared - system reset.`
	
	If you use "vmcp" to address CP:
	`vmcp define storage 512m`
	`STORAGE = 512M`
	`Storage cleared - system reset.`

Define additional virtual CPUs	DEFINE CPU *cpuaddr*
	
	Examples:
	`%cp define cpu 1`
	`CPU 01 DEFINED`
	
	If you use "vmcp" to address CP:
	`vmcp define cpu 1`
	`CPU 01 DEFINED`

Display the status of the cryptographic facility	Q V CRYPTO
	
	Examples:
	`%cp q v crypto`
	`AP 63 CEX2A Queue 01 shared`
	`Ready;`
	
	If you use "vmcp" to address CP:
	`vmcp q v crypto`
	`AP 63 CEX2A Queue 01 shared`
	`Ready;`

If you want to ...	Then use this command ...

Detach virtual CPUs and devices from the virtual machine	DETACH
	
	Examples:
	• Virtual CPUs:
	Attention: Issue this command only when Linux is shut down because detaching virtual CPUs resets the virtual machine and any operating system running in it.
	`%cp detach cpu 01`
	`00: CPU 01 detached`
	`00: Storage cleared - system reset.`
	If you use "vmcp" to address CP:
	`vmcp detach cpu 01`
	`00: CPU 01 detached`
	`00: Storage cleared - system reset.`
	• Minidisks:
	`%cp detach 200`
	`DASD 0200 DETACHED`
	If you use "vmcp" to address CP:
	`vmcp detach 200`
	`DASD 0200 DETACHED`
	• Expanded storage:
	`%cp detach xstore`
	If you use "vmcp" to address CP:
	`vmcp detach xstore`
	• Guest LAN segments:
	`%cp detach lan qdio0`
	`LAN VMUSERX QDIO0 is destroyed`
	If you use "vmcp" to address CP:
	`vmcp detach lan qdio0`
	`LAN VMUSERX QDIO0 is destroyed`

If you want to ...	Then use this command ...

Link to another disk (read-only mode)	LINK *userid vdev1 vdev2* RR
	
	vdev1 is *userid*'s virtual address; *vdev2* is an available virtual address in your virtual machine configuration.
	
	Examples:
	`%cp link linux01 150 150 rr`
	
	If you use "vmcp" to address CP:
	`vmcp link linux01 150 150 rr`

Load (boot) Linux in a virtual machine	IPL
	
	IPL (initial program load) and boot are synonymous.
	
	Examples:
	`%cp ipl 150 clear`
	
	If you use "vmcp" to address CP:
	`vmcp ipl 150 clear`

Make CP display the HOLDING status when the display screen is full and highlighted output appears on the screen.	TERMINAL HOLD ON
	
	When the display screen is in HOLDING status and you want to clear the display, press the PA1 or CLEAR key.
	
	Examples:
	`%cp terminal hold on`
	
	If you use "vmcp" to address CP:
	`vmcp terminal hold on`

Change the number of seconds that elapse between the time when CP issues the MORE... state and sounds the terminal alarm, and between the time when CP sounds the alarm and clears the display.	TERMINAL MORE *mm nn*
	
	Examples:
	`%cp terminal more 5 2`
	
	If you use "vmcp" to address CP:
	`vmcp terminal more 5 2`
	
	means CP waits 5 seconds after issuing the MORE... state before sounding the terminal alarm, then 2 seconds after sounding the alarm before clearing the display.

You are done.

Archiving and backing up critical data

Archiving stores large bodies of data (for example, an entire disk image) for safekeeping, and should be a part of your disaster recovery plan. The data should be mutually consistent, so you can be running, but cannot be making changes. Do archiving on regular intervals, such as weekly or monthly, or whenever you do major software changes. These archives allow you to restore entire systems quickly.

In contrast to archiving, backups store incremental changes, such as changes to individual files, and you can do backups during system operations. Backups are part of an on-demand file-level recovery system and you should do backups daily. To restore backups, you need a running system, so after a system disaster, use your archive to restore the entire system, then use your backups to restore files.

Sometimes the terms "archive" and "backup" are used to mean the same thing. This topic uses the term "archive" and describes how to archive large bodies of data such as disks. For backups on z/VM, use an incremental backup system, such as Tivoli® Storage Manager. For incremental backup procedures, consult the documentation for your backup product.

On z/VM, archive:

- The z/VM system disks
- The z/VM spooling system, especially named save segments (NSSs) and discontiguous saved segments (DCSSs)
- Each major Linux kernel or installed software change.

It is not necessary to archive the z/VM paging space, since these disks hold data valid only for the currently-running z/VM system.

z/VM provides two service programs for archiving:

- The DASD Dump Restore (DDR) utility program allows you to create archives of minidisks and complete DASD volumes. The program does not do incremental backups, so all data on a disk is archived whether or not it has changed. There are two versions of the program: one is the DDR command, which you can issue from CMS; the other is a standalone program that you can load (IPL).
- SPXTAPE produces an archive of spool files. Since NSSs (like CMS) and DCSSs are part of the spooling system, you should archive the spooling system. If problems develop with the spooling system and you need to do a CLEAN start of z/VM, it is much easier to restore archived NSSs and DCSSs instead of rebuilding them.

Related information:

- "DDR" in *z/VM: CP Commands and Utilities Reference*, SC24-6175
- "SPXTAPE" in *z/VM: CP Commands and Utilities Reference*, SC24-6175

Overview of archiving z/VM system data

At installation time, you should have archived the z/VM 610RES, 610W01, and 610W02 system disks. During the procedure, you created a tape that has the DDRXA program and a copy of all the system disks. DDRXA is a standalone utility program that you can load (IPL) from the tape, then use to restore the system to disk. Because DDRXA does not require an operating system, the archive tape is useful for disasters and disaster tests in which z/VM must be restored.

Also at installation time, you used the SPXTAPE command to archive name saved systems, for example CMS, and discontiguous saved segments.

Table 7 directs you to procedures to follow for archiving the z/VM system disks, named saved systems, and discontiguous saved segments.

Table 7. Task roadmap for archiving system data

Subtask	Associated instructions (see . . .)
Archiving named saved systems and discontiguous saved segments	"Step 6. Back Up the Named Saved Systems and Segments" in *z/VM: Guide for Automated Installation and Service*, GC24-6197
Archiving the z/VM system disks	"Step 7. Store a Backup Copy of the z/VM System on Tape" in *z/VM: Guide for Automated Installation and Service*, GC24-6197

Archiving virtual server disks

You should archive your Linux virtual server system minidisks after you install Linux or make a software change, such as adding an application. By archiving the system minidisks (for instance, the 150 and 151 disks), you have the ability to restore your Linux operating system quickly in case of an emergency.

To archive your virtual server minidisks, use the DDR command in CMS. The DDR command requires CMS and a running z/VM system. If you do not have a running z/VM system, restore it from the loadable tape you created as an archive of z/VM (see "Overview of archiving z/VM system data" on page 112).

Because the data on the minidisks must be mutually consistent, you cannot be changing the minidisk contents while you archive. Archiving is best done while the virtual server is down.

Steps for archiving virtual server disks

Before you begin: Shut down the Linux virtual server. You need to log onto MAINT.

Perform these steps to archive virtual server disks:

1. Mount a scratch tape on a tape drive.
2. Attach the tape to MAINT. Type this command and press the Enter key:

   ```
   attach rdev maint 181
   ```

   where *rdev* is the real device address of the tape drive.
3. Link to the Linux virtual server disk you want to archive. From the command line, type this command and press the Enter key:

   ```
   link userid vdev1 vdev2 rr
   ```

   where *vdev1* is the virtual device address of a Linux minidisk and *vdev2* is an available virtual device address in MAINT's configuration.

   Example: To link MAINT to LINUX01's 150 minidisk, issue this command:

   ```
   link linux01 150 950 rr
   ```

4. From the CMS command line, type the DDR command and press the Enter key:

```
ddr
z/VM DASD DUMP/RESTORE PROGRAM
ENTER:
```

   DDR prompts for control statements by issuing the ENTER: prompt.

5. At the ENTER: prompt, type this command and press the Enter key:

```
sysprint cons
```

6. Designate the disk from which DDR dumps the data. Type this command and press the Enter key:

```
input vdev2 dasd
```

   where *vdev2* is MAINT's virtual device address (corresponding to the Linux virtual server disk).

   Example:

```
input 950 dasd
```

7. Designate the tape to which the data is to be dumped. Type this command and press the Enter key:

```
output 181 tape scratch
```

8. Dump the disk to tape. Type this command and press the Enter key, then respond to the prompts as shown:

```
dump all
HCPDDR711D VOLID READ IS LIN150
DO YOU WISH TO CONTINUE?  RESPOND YES, NO OR REREAD:
yes
⋮
DUMPING   LIN150
⋮
ENTER:
```

9. To exit DDR, press the Enter key.
10. Repeat these steps for each virtual server disk you want to archive.

Continue with the next procedure.

Steps for restoring virtual server disks

You should test the restoration procedure so you know what to expect in case you have a system disaster. To test the restoration, use a test Linux virtual server with disk sizes the same as your archived Linux disks. When you are satisfied with the restoration procedure, you can remove the test Linux virtual server.

Before you begin: You need a test Linux virtual server. Follow the steps to clone a virtual server in Chapter 8, "Cloning Linux virtual servers," on page 83. The test Linux virtual server must be shut down and logged off. You need to log onto MAINT.

Perform these steps to restore virtual server disks:

1. Mount the archive tape on a tape drive.
2. Attach the tape to MAINT. Type this command and press the Enter key:

   ```
   attach rdev maint 181
   ```

   where *rdev* is the real device address of the tape drive.
3. Link to the test Linux virtual server disk to which you want restore the archive. From the command line, type this command and press the Enter key:

   ```
   link userid vdev1 vdev2 w
   ```

   where *vdev1* is the virtual device address of the Linux minidisk and *vdev2* is an available virtual device address in MAINT's configuration.

   Example: To link MAINT to LINTEST's 150 minidisk, issue this command:

   ```
   link lintest 150 950 w
   ```
4. From the CMS command line, type the DDR command and press the Enter key:

   ```
   ddr
   z/VM DASD DUMP/RESTORE PROGRAM
   ENTER:
   ```

   DDR prompts for control statements by issuing the ENTER: prompt.
5. At the ENTER: prompt, type this command and press the Enter key:

   ```
   sysprint cons
   ```
6. Designate the tape from which DDR restores the data. Type this command and press the Enter key:

   ```
   input 181 tape
   ```
7. Designate the disk to which the data is to be restored. Type this command and press the Enter key:

   ```
   output vdev2 dasd
   ```

   where *vdev2* is MAINT's virtual device address (corresponding to the test Linux virtual server disk).

   Example:

   ```
   output 950 dasd
   ```

8. Restore the tape to disk. Type this command and press the Enter key, then respond to the prompts as shown:

```
restore all
HCPDDR711D VOLID READ IS LIN150
DO YOU WISH TO CONTINUE?  RESPOND YES, NO OR REREAD:
yes
.
.
.
RESTORING   LIN150
.
.
.
ENTER:
```

9. To exit DDR, press the Enter key.
10. Repeat these steps for each virtual server disk you want to restore.
11. Log onto the test Linux virtual server to see if it initializes properly.

You know you are done when the restoration is successful.

Chapter 11. Monitoring performance and capacity

This topic explains how to monitor z/VM functions such as paging and spooling space. The topic includes run-time characteristics and symptoms you should observe to avoid system problems.

Overview of performance monitoring

To be successful in monitoring and tuning your z/VM system, you first need to decide what the phrase "good performance" means for your installation. What is considered good performance varies from one installation to another. For example, for some installations response time for commands is the primary indicator of performance, and good performance might mean sub-second response times. You need to evaluate your own workload, decide what the best indicators of performance are for your workload, and decide what values for those indicators constitute good performance in your environment.

Tuning a z/VM system is really nothing more than matching your hardware's capabilities to your workload's characteristics and to your definition of good performance. Through routine monitoring, you can keep track of the characteristics of your workload, the utilization of your hardware, and, most important, whether your definition of good performance is being met.

As you work at tuning your z/VM system, keep in mind that the changes and adjustments you make serve only to get the best performance from the physical resources you have. Sometimes there is nothing more you can do for performance, due to the limitations of your hardware; when all else fails, you need to consider purchasing more.

Because performance ultimately depends on the hardware available, it's important that you, the performance analyst, have an accurate inventory of the physical resources present on your machine. In particular, as you consider tuning your z/VM system, have an accurate inventory of the hardware available to run your workload:

- How many real CPUs are there, and how fast are they?
- How much real storage is available? How is that real storage distributed between central storage and expanded storage (XSTORE)?
- How well is the system equipped for paging?
  - How much total paging space is there?
  - How many paging volumes are there?
  - How are the paging volumes distributed among disk controllers?
  - How fast are the paging volumes and their disk controllers?
  - How many channel paths (CHPIDs) are there from the processor to each disk controller used in paging?
  - For each volume used for paging, are there other kinds of data on the volume besides paging space?
- How well is the system equipped for spooling? The paging questions apply to the spooling configuration, too.

Tuning is the art of optimizing some performance measure of a workload within the constraints imposed by the hardware available. Because the hardware always

imposes some type of constraint, tuning often becomes a balancing act, trading off the needs of the whole system against the needs of specific virtual machines. Sometimes providing more resources to one virtual machine is detrimental to other virtual machines or to the system as a whole. For instance, you might not want virtual machines used for testing programs to get as much system resources as production virtual machines, because your objective is to give production work maximum throughput. Performance monitoring can help you understand which changes meet your performance objectives.

You can use the INDICATE command (class B, C, and E users) to obtain an informal snapshot of system load, scheduler dispatcher queues, and I/O. The QUERY SHARE command (class A and E) allows you to view the share of system resources a virtual machine has. Both of these commands provide only a snapshot, while the MONITOR command collects large amounts of performance measurement data for later systematic and comprehensive analysis.

The MONITOR command (class A and E users) starts and stops the emission of data relevant to specific system events. The command also starts and stops the collection, and periodic emission, of sample data descriptive of system performance.

Monitor *domains* divide the emitted data into topic areas: behavior of I/O, behavior of the scheduler, configuration of the z/VM system, settings of the monitor itself, and so on. Data the monitor emits under a certain domain includes both event information (such as a user logoff) and sample information (collected performance measurement data, usually counters).

Monitoring Linux virtual servers with Performance Toolkit for VM

z/VM provides analysis tools, such as Performance Toolkit for VM, that helps you analyze the data you collect with the MONITOR command. In addition to analyzing z/VM performance data, Performance Toolkit for VM processes Linux performance data, provided you have the proper support software. To process Linux performance data, you have these choices:

- Install a commercial Linux on a system that contains a mainframe performance monitoring product.
- Use a Linux 2.6 kernel such as SLES9, which has built-in support that allows Performance Toolkit for VM to monitor Linux virtual servers.
- For additional performance data from Linux, install RMF™ Performance Monitor and configure Performance Toolkit for VM to access this data.

  Note: The RMF Performance Monitor is produced by an IBM Technical Study and is not a part of any IBM product.

Related information

- For more about z/VM performance, see *z/VM: Performance*, SC24-6208.
- For more about Performance Toolkit for VM, see *z/VM: Performance Toolkit Reference*, SC24-6210.

Overview of the z/VM scheduler

This topic gives a rudimentary introduction to the z/VM scheduler so you can understand system responses to commands such as INDICATE.

The z/VM scheduler controls the dispatching of virtual machines by managing three scheduling lists:

- The *dormant list* contains a list of virtual machines that are idle or waiting for completion of a long event, such as a tape read.
- The *eligible list* contains a list of virtual machines waiting for resources. Each virtual machine is classified according to its processing characteristics (known as its *transaction class*). A scheduler *transaction* is the amount of time the virtual machine remains able to run. A virtual machine that runs short transactions consumes little processor resource between waits, while one that runs long transactions consumes a large amount of processor resource between waits. The transaction classes are:
  - E0. Virtual machines in this class do not wait in the eligible list, but move to the dispatch list immediately.
  - E1. Virtual machines in this class are those doing short transactions, such as interactive users.
  - E2. Virtual machines in this class do medium-length transactions.
  - E3. Virtual machines in this class do long transactions.

  The scheduler assesses each virtual machine for its need for available resources. For the scheduler, *resources* include processors, central storage, and paging capacity. When a virtual machine is waiting for resources, it is waiting for the z/VM scheduler to decide that there is enough processor capacity, storage capacity, and paging capacity to add this virtual machine to the set of virtual machines on the dispatch list.

  The scheduler calculates the eligible priority of a virtual machine when it enters the eligible list. This priority is called the *deadline priority*, the time by which the virtual machine should enter the dispatch list. The relative priorities of virtual machines are designed to slow down virtual machines that require highly-demanded resources, grant virtual machines their shares of available system resources, and control the amount of resource each class gets. For instance, though E2 and E3 virtual machines wait longer in the eligible list, they receive longer elapsed time slices in the dispatch list, which allows efficient use of system resources and the rapid completion of interactive work.
- The *dispatch list* contains a list of virtual machines that can run or whose wait state is expected to be short (for instance, waiting for a page-in). When a virtual machine enters the dispatch list, it retains its transaction class; E0 virtual machines become Q0, E1 become Q1, and so forth. Also, each virtual machine gets a *deadline priority*, the time of day when the virtual machine should complete its next dispatch time slice under ideal conditions. The lower the dispatch priority, the closer the virtual machine is to being dispatched.

  Because dispatching priorities are dynamically calculated, the sequence of the virtual machines in the dispatch list varies according to the changing operating characteristics of the virtual machines.

The scheduler controls the cycling of virtual machines through the three lists. When a virtual machine logs on, it is placed in the dormant list and moved to the eligible list only when it has work to do. When entering the eligible list, the virtual machine is assigned its deadline priority based on its share, resource requirement, and contention for system resources. When resources become available, the scheduler moves virtual machines from the eligible list to the dispatch list and assigns them dispatch priorities. As virtual machines consume CPU time in the dispatch list, they are examined and reassigned priority as their dispatch time slices end. Because a virtual machine consumes a given amount of processing or storage resource, becomes idle, or is preempted in favor of other virtual machines

in the eligible list, it moves back to the eligible list (if it can still be dispatched) or the dormant list (if it can no longer be dispatched).

Related information

For a complete description of the scheduler, see "Virtual Machine Scheduling and Dispatching" in *z/VM: Performance*, SC24-6208.

Steps for taking a snapshot of system performance

Before you begin: You need class E authority. You need to understand scheduler basics and the scheduler lists. See "Overview of performance monitoring" on page 117.

Perform these steps to take a snapshot of system performance:

1. Log on a user ID with class E authority.
2. Base your action on this table:

If you want to check ...	Then issue ...

System load	**cp indicate load**
	
	Example:
	`cp indicate load`
	`AVGPROC-061% 03`
	`XSTORE-000001/SEC MIGRATE-0000/SEC`
	`MDC READS-000028/SEC WRITES-000001/SEC HIT RATIO-100%`
	`STORAGE-015% PAGING-0006/SEC STEAL-000%`
	`Q0-00002(00000)  DORMANT-00487`
	`Q1-00001(00000)  E1-00000(00000)`
	`Q2-00001(00000) EXPAN-002 E2-00000(00000)`
	`Q3-00009(00000) EXPAN-002 E3-00000(00000)`
	`PROC 0000-057%  PROC 0001-060%`
	`PROC 0002-067%`
	`LIMITED-00000`
	
	Notes:
	1. AVGPROC indicates utilization of all CPUs
	2. MIGRATE=0 means all working sets of the logged-on virtual machines fit in central storage and expanded storage. A growing MIGRATE number indicates that central storage and expanded storage are insufficient to contain the working sets of the logged-on virtual machines, so the system has to resort to paging to DASD. The higher the MIGRATE number is, the greater is your workload's paging demands on the amount of real storage (central plus expanded) on your system.
	3. STORAGE indicates the real storage utilization.
	4. Q0 — Q3 are virtual machines in the dispatch list.
	5. E1 — E3 are virtual machines in the eligible list.
	6. DORMANT indicates virtual machines that are dormant.

If you want to check ...	Then issue ...

Virtual machines in the dispatch and eligible lists	**cp indicate queues exp**
	
	Example:
	
	`cp indicate queues exp`
	`TCPIP Q0 PS 00006309/00005795 .... -101.6 A02`
	`BKW Q1 R00 00001339/00001352 .I.. -91.50 A00`
	`RSCS Q0 PG 00000614/00000613 .... -1.227 A02`
	`TKNAIRB Q3 R 00016601/00016504 ..D. .0084 A02`
	`REFSNID Q3 R01 00001708/00001542 .... .0601 A01`
	`FARMAN Q3 IO 00003528/00003368 ..D. .2457 A02`
	`EDLLNX2 Q3 PS 00017257/00017144 .... 99999 A02`
	`VTAM Q0 PS 00001865/00001864 .I.. 99999 A02`
	`SSLSERV Q3 PS 00000374/00000345 .... 99999 A02`
	`VMLINUX1 Q3 PS 00000987/00000987 ..D. 99999 A02`
	`VMLINUX Q3 PS 00002854/00002854 .... 99999 A02`
	`EDLLNX1 Q3 PS 00005165/00005165 .... 99999 A02`
	`CORAK2 Q3 PS 00147973/00147973 .... 99999 A02`
	
	Notes:
	
	1. Virtual machines are listed in priority order.
	2. When INDICATE QUEUES EXP consistently shows E1, E2, or E3 (eligible) users, an eligible list is forming because one or more system resources appears to be constrained.
	3. The third column explains why a virtual machine is in a wait state:
	• R*nn* — current RUNUSER on the specified real CPU, where *nn* is the CPU ID (in hexadecimal)
	• IO — waiting for I/O
	• PS — PSW wait (enabled wait state)
	• PG — waiting for paging

I/O	**cp indicate i/o** (several times to see a pattern)
	
	Example:
	
	`cp indicate i/o`
	`FCONX 73DC TCPIP310 ---- VMBACKUP ---- TCPIP ---- MONWRITE 7389`
	`RTMTEST 73DF TOMDEF 80C9`
	
	Notes:
	
	1. INDICATE I/O shows virtual machines currently in an I/O wait state and also shows the real device number to which the most recent I/O operation was mapped. Four dashes (----) indicates a virtual device.
	2. If you see the same real device number for several virtual machines, this might be an indication that there are many minidisks on the same DASD or you have a DASD controller bottleneck, where too many DASD doing I/O are on the same controller.

If you want to check ...	Then issue ...

Whether a virtual machine is hung	**cp indicate user** *userid* **exp** (several times at 10–second intervals)
	
	Example:
	`cp indicate user bkw exp`
	`Userid=BKW  Mach=XC  V=V  Attached xstore=NONE`
	`Iplsys=CMS  Devnum=27`
	`Spool:  Reads=335  Writes=129`
	`Owned spaces:  Number=1  Owned size=247M`
	`  Primary space:  ID=BKW:BASE  PRIVATE`
	`    Defined size=256M  Address limit=814M`
	`  Private spaces:  Number=1  Owned size=247M`
	`    Pages:  Main=1369  Xstore=2  Dasd=0`
	`            Locked=0  WS=1349  Reserved=0`
	`  Shared spaces:   Number=0  Owned size=0`
	`    Pages:  Main=0  Xstore=0  Dasd=0`
	`            Locked=0`
	`  Private paging:`
	`    Xstore:  Reads=8          Writes=10         Migrates=0`
	`    Dasd:    Reads=0          Writes= 0`
	`  Shared paging:`
	`    Xstore:  Reads=0          Writes= 0         Migrates=0`
	`    Dasd:    Reads=0          Writes= 0`
	`CPU 00: Ctime=0 07:36:37  Vtime=0 00:00:02  Ttime=0 00:00:03`
	`       Rdr=404  Prt=0  Pch=16  IO=3970`
	
	Notes:
	1. If you do not see VTIME, TTIME, and IO increasing, the virtual machine might be in a legitimate enabled wait state, but this also indicates the virtual machine might need to be restarted. Warn users and contact the virtual machine owner before you restart the virtual machine.

A virtual machine's share of resources	**cp query share** *userid*
	
	Example: Assume your installation has four active virtual machines.
	`cp query share linux0`
	`LINUX0  : ABSOLUTE SHARE = 40%`
	`           MAXIMUM SHARE = NOLIMIT`
	
	`cp query share linux1`
	`LINUX1  : RELATIVE SHARE = 200`
	`           MAXIMUM SHARE = NOLIMIT`
	
	`cp query share linux2`
	`LINUX2  : RELATIVE SHARE = 100`
	`           MAXIMUM SHARE = NOLIMIT`
	
	`cp query share linux3`
	`LINUX3  : RELATIVE SHARE = 100`
	`           MAXIMUM SHARE = NOLIMIT`
	
	Notes:
	1. A virtual machine receives its proportion of any scarce resource (CPUs, real storage, or paging I/O capability) according to its SHARE setting.
	2. LINUX0 has an absolute 40% share and therefore receives up to approximately 40% of available resources.
	3. LINUX1 receives up to 200/400 (LINUX1's relative share divided by the total relative shares) of the remaining 60% of available resources.
	4. LINUX2 and LINUX3 each receive up to 100/400 of the remaining 60% of available resources.

If you want to check ...	Then issue ...

Paging	**cp query alloc page**
	
	Example:
	
	`cp query alloc page`
	`                EXTENT     EXTENT  TOTAL  PAGES   HIGH    %`
	`VOLID  RDEV      START        END  PAGES IN USE   PAGE USED`
	`------ ---- ---------- ---------- ------ ------ ------ ----`
	`K4E40A C621          1       3338 600840   1846   1898   1%`
	`K41006 1006          1       3338 600840   1335   1375   1%`
	`K41007 1007          1       3338 600840   1839   1890   1%`
	`K4PAG1 DB3B          1       3338 600840   2442   2462   1%`
	`K4PAG2 DB3C          1       3338 600840   2209   2247   1%`
	`K4PAG3 DB3D          1       3338 600840   2340   2394   1%`
	`K4PAG4 DB3E          1       3338 600840   2621   2621   1%`
	`K4TDSK DA06          0          0    180    180    180 100%`
	`                                  ------ ------         ----`
	`SUMMARY                            4107K  14812          1%`
	`USABLE                             4107K  14812          1%`
	`Ready;`

Spooling	**cp query alloc spool**
	
	Example:
	
	`cp query alloc spool`
	`                EXTENT     EXTENT  TOTAL  PAGES   HIGH    %`
	`VOLID  RDEV      START        END  PAGES IN USE   PAGE USED`
	`------ ---- ---------- ---------- ------ ------ ------ ----`
	`K4551C 8B2E          1       3338 600840 599500 600840  99%`
	`K4551D 8B2F        502       3338 510660 508033 510660  99%`
	`K4961C 961C        501       2654 323100 321442 323100  99%`
	`K4E504 C021          1       3338 600840  43651  43651   7% DUMP`
	`K47808 7808          0       3338 601020  81642 600988  13%`
	`K4951D 951D          0       3338 601020 588935 601020  97%`
	`                                  ------ ------         ----`
	`SUMMARY                            3162K  2093K          66%`
	`USABLE                             2575K  2050K          79%`
	`Ready;`

You are done.

Using the CP Monitor and Performance Toolkit for VM

This topic has:

- An overview of the CP Monitor and Performance Toolkit for VM
- Configuring Performance Toolkit for VM
- Using Performance Toolkit for VM to analyze performance and capacity.

Overview of the CP Monitor and Performance Toolkit for VM

The CP Monitor collects system performance data. This data can be processed by an external data reduction program to produce formatted reports or real-time displays that give you an understanding of system operation or help you analyze the use of, and contention for, major system resources. These resources include CPUs, storage, I/O devices, and the paging subsystem. You can control the amount and nature of the data collected. As Figure 10 on page 124 shows, monitoring is in this order:

Stage	Description
1	You use the CP privileged command MONITOR to control monitoring, including the type, amount, and nature of data to be collected.
2	An application program running in a CMS virtual machine connects to the CP *MONITOR System Service to establish a data link with CP.
3	The monitor collects performance data during CP operation and stores it, in the form of monitor records, in a saved segment (MONDCSS). **Note:** MONDCSS is already defined for you during system installation.
4	An IBM-supplied program called MONWRITE retrieves monitor records from the saved segment and processes them. MONWRITE can store monitor records on disk or tape.
5	An application program, such as Performance Toolkit for VM, can read data from the saved segment to display real-time performance characteristics. Performance Toolkit for VM can also process monitor data records written to disk or tape by MONWRITE. While processing monitor data, Performance Toolkit for VM can generate historical monitor data that it can process at a later time.

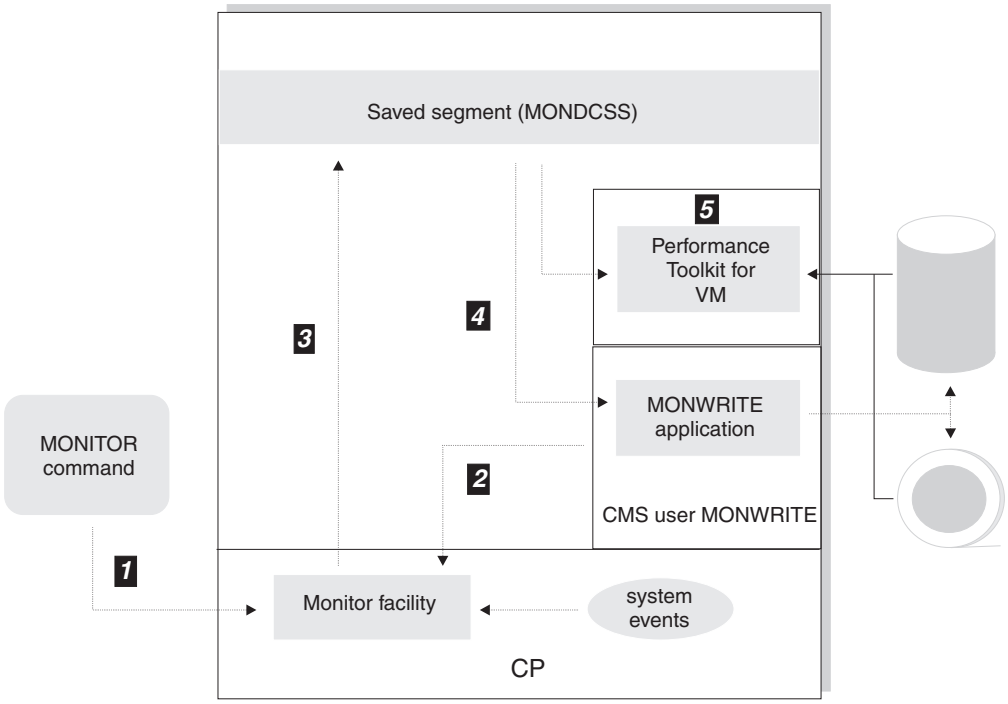

Figure 10. General monitoring process

Configuring Performance Toolkit for VM

Though Performance Toolkit for VM comes with z/VM, it is an optional feature in a disabled state. If you decide to use this feature, you must place an order for it so that you can enable it. This document assumes you have installed Performance Toolkit for VM according to *Program Directory for Performance Toolkit for VM*. Additional setup is required.

To set up Performance Toolkit for VM, you must perform these subtasks:

Subtask	Associated instructions (see . . .)
Setting up the monitoring virtual machine program (MONWRITE)	"Steps for setting up the monitoring virtual machine (MONWRITE)"
Setting up the performance analysis virtual machine (PERFSVM)	
• Installing Performance Toolkit for VM	• *Program Directory for Performance Toolkit for VM* at http://www.ibm.com/eserver/zseries/zvm/library/
• Configuring Performance Toolkit for VM	• "Steps for configuring Performance Toolkit for VM" on page 126 • "Steps for checking your Performance Toolkit for VM configuration" on page 129
(Optional) Setting up your Linux virtual servers to be monitored by Performance Toolkit for VM **Note:** To monitor your Linux virtual servers through Performance Toolkit for VM, you might need to install additional software on your Linux systems. See "Monitoring Linux virtual servers with Performance Toolkit for VM" on page 118.	"(Optional) Steps for setting up your Linux virtual servers to be monitored by Performance Toolkit for VM" on page 130
(Optional) Setting up the Web interface for Performance Toolkit for VM	"(Optional) Steps for setting up the Web interface for Performance Toolkit for VM" on page 131
(Optional) Testing your monitoring setup	"(Optional) Steps for testing your Web interface setup" on page 132

Steps for setting up the monitoring virtual machine (MONWRITE)

Use the default MONWRITE virtual machine to collect monitor data and to write the data to disk using the MONWRITE program. These steps show you how to set up the MONWRITE virtual machine so that the correct commands are issued automatically when MONWRITE logs on.

Before you begin: You need log onto the MONWRITE virtual machine.

Perform these steps to set up the monitoring virtual machine:

1. Issue the BEGIN command to remove MONWRITE from CP READ state. From the command line, type this command and press the Enter key:

```
begin
```

2. Edit the PROFILE EXEC for the MONWRITE virtual machine. Type this command and press the Enter key:

```
xedit profile exec a
```

3. Type these lines in the PROFILE EXEC:

```
 PROFILE  EXEC     A1  V 130  Trunc=130 Size=28 Line=4 Col=1 Alt=0

* * * Top of File * * *
/**/
'SET RUN ON'
'SET PF12 RETRIEVE'
'CP MONITOR EVENT ENABLE ALL'
'CP MONITOR SAMPLE ENABLE ALL'
'CP MONITOR START'
'CP MONITOR EVENT DISABLE SEEKS ALL'
'CP MONITOR EVENT DISABLE USER ALL'
'CP MONITOR EVENT DISABLE SCHEDULER ALL'
'MONWRITE MONDCSS *MONITOR DISK'
EXIT
```

Tips:

- SEEKS, USER, and SCHEDULER are events most installations do not record. If you do not disable these events, you need a larger MONDCSS saved segment.
- The MONWRITE command invokes the MONWRITE program and sends monitor data to a CMS file whose file name has this pattern:

  D*date* T*time* A1

  Where *date* is the current date and *time* is the current time. This is handy if you want to stop and restart MONWRITE, which creates individual CMS files for each invocation. For instance, you can stop MONWRITE at the end of the day and start it up again in the morning. Watch the percentage of the disk used due to monitor data being stored in this file so the disk does not fill up. To check the disk, use the CMS QUERY ACCESSED command.

4. Save the PROFILE EXEC. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

5. Check that AUTOLOG1's PROFILE EXEC automatically logs MONWRITE on. See the instructions in "Steps for automatically starting Linux virtual servers and other virtual machines" on page 85.

6. Test the PROFILE EXEC. From the CMS command line, issue:

```
profile
HCPMOW6265A MONITOR WRITER CONNECTED TO *MONITOR, START CP MONITOR
```

7. Disconnect MONWRITE. From the command line, type this command and press the Enter key:

```
%cp disc
```

You know you are done when the PROFILE EXEC runs successfully and you disconnect MONWRITE.

Steps for configuring Performance Toolkit for VM

Before you begin: You need to install Performance Toolkit for VM by following the *Program Directory for Performance Toolkit for VM* (http://www.ibm.com/eserver/zseries/zvm/library/).

You need to log on to PERFSVM.

Perform these steps to configure Performance Toolkit for VM:

1. Issue the BEGIN command to remove PERFSVM from CP READ state. From the command line, type this command and press the Enter key:

```
begin
```

2. Remove the comments from the MONITOR commands in PERFSVM's PROFILE EXEC.
   a. Edit the PROFILE EXEC. From the command line, type this command and press the Enter key:

```
xedit profile exec a
```

   b. Remove the "/*" and "*/" characters that surround the MONITOR commands you need.

      Example: This PROFILE EXEC for PERFSVM enables monitor sampling for the typical samples and events that you need:

```
 PROFILE  EXEC     A1  V 130  Trunc=130 Size=12 Line=5 Col=1 Alt=9

00000 * * * Top of File * * *
00001 'CP MONITOR SAMPLE DISABLE ALL'
00002 'CP MONITOR EVENT DISABLE  ALL'
00003 'CP MONITOR SAMPLE ENABLE PROCESSOR'
00004 'CP MONITOR SAMPLE ENABLE STORAGE'
00005 'CP MONITOR SAMPLE ENABLE NETWORK'
00006 'CP MONITOR SAMPLE ENABLE USER ALL'
00007 'CP MONITOR SAMPLE ENABLE I/O ALL'
00008 'CP MONITOR SAMPLE ENABLE APPLDATA ALL'
00009
00010 'CP MONITOR EVENT  ENABLE STORAGE'
00010 'CP MONITOR EVENT  ENABLE I/O ALL'
00011
00012 'CP MONITOR sample INTERVAL 1 MIN'
```

   c. Save the file. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

3. Copy the FCONX $PROFILE to the 191 A-disk. Issue this command:

```
copy fconx $profile d = = a
Ready;
```

4. Edit FCONX $PROFILE A.
   a. From the command line, type this command and press the Enter key:

```
xedit fconx $profile a
```

   b. Find the line with "MONCOLL VMCF". From the XEDIT command line, type this command and press the Enter key:

```
====> /moncoll vmcf
```

c. Replace the asterisk (*) in column 1 with an "F" for the "MONCOLL VMCF" statement:

```
FC MONCOLL VMCF ON
```

d. Save the FCONX $PROFILE. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

5. Create an FCONRMT SYSTEMS file that links your z/VM system and PERFSVM to a special resource name called FCXRES00.

Note: FCXRES00 is a special resource name used for remote access. This name is already defined for you; however, you have to make it known by updating FCONRMT SYSTEMS.

a. Edit FCONRMT SYSTEMS A. From the command line, type this command and press the Enter key:

```
xedit fconrmt systems a
```

b. Enter input mode. From the XEDIT command line, type this command and press the Enter key:

```
====> input
```

c. Type this line and press the Enter key twice:

```
 FCONRMT  SYSTEMS  A1  F 80  Trunc=80 Size=1 Line=1 Col=1 Alt=1
DMSXMD587I XEDIT:



* * * Top of File * * *
node_id  PERFSVM   ESA  Y    FCXRES00
```

where *node_id* is your z/VM system identifier.

d. Save the file. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

6. Create an FCONRMT AUTHORIZ file that makes PERFSVM a store and forward server. Additionally, configure PERFSVM to allow it to issue CP and CMS commands.

a. Edit FCONRMT AUTHORIZ A. From the command line, type this command and press the Enter key:

```
xedit fconrmt authoriz a
```

b. Enter input mode. From the XEDIT command line, type this command and press the Enter key:

```
====> input
```

c. Type these lines and press the Enter after each line:

```
 FCONRMT  AUTHORIZ A1  F 80  Trunc=80 Size=18 Line=2 Col=1 Alt=2
DMSXMD573I Input mode:


* * * Top of File * * *
node_id PERFSVM  S&FSERV  DATA
node_id PERFSVM  CMD
node_id  * CMD DATA
```

where *node_id* is your z/VM system identifier.

d. Press the Enter key once more, then save the file. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

7. Check that AUTOLOG1's PROFILE EXEC automatically logs PERFSVM on. See the instructions in "Steps for automatically starting Linux virtual servers and other virtual machines" on page 85.

Stay logged on to PERFSVM and continue with the next instructions.

Steps for checking your Performance Toolkit for VM configuration

Before you begin: You need to be logged onto PERFSVM.

Perform these steps to check the configuration:

1. From the command line, run PERFSVM's PROFILE EXEC to start Performance Toolkit for VM. From the command line, type this command and press the Enter key:

```
profile
```

Result: You see the basic mode screen.

```
FCX001                Performance Toolkit for VM               Autoscroll 12
 FCXBAS500I Performance Toolkit for VM FL610 BASE
 FCXAPP530I Connected to *IDENT for resource FCXRES00
 FCXAPF530I Connected to *IDENT for resource FCXSYSTM
 HCPMOF6229E Monitor event collection is already active.
 HCPMOG6229E Monitor sample collection is already active.
```

2. From the basic mode screen, issue this command:

```
monitor
```

Result:

```
FCX124          Performance Screen Selection  (FL610 PITS353)    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. Storage subpools        13. I/O device load*       33. Benchmark displays*
4. Priv. operations        14. CP owned disks*        34. Correlation coeff.
5. System counters         15. Cache extend. func.*   35. System summary*
6. CP IUCV services        16. DASD I/O assist        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. Paging activity
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
Select performance screen with cursor and hit ENTER
Command ===>
F1=Help  F4=Top  F5=Bot  F7=Bkwd  F8=Fwd  F12=Return
```

3. Wait to allow Performance Toolkit for VM to collect data. Performance Toolkit for VM will not perform most screen calculations until at least two monitor intervals have passed. At least two monitor intervals are required to provide the delta for certain fields. When available, options are highlighted.
4. Press the PF12 key to return to basic mode.

You are done when you know Performance Toolkit for VM is working. If you plan to follow the next instructions, stay logged onto PERFSVM.

(Optional) Steps for setting up your Linux virtual servers to be monitored by Performance Toolkit for VM

Before you begin: You need to be logged on to PERFSVM. You need to have the appropriate monitoring software installed on your Linux operating systems (see "Monitoring Linux virtual servers with Performance Toolkit for VM" on page 118).

Perform these steps to set up your Linux virtual servers to be monitored by Performance Toolkit for VM:

1. Edit the FCONX LINUXUSR file. Issue this command:

```
xedit fconx linuxusr a
```

2. Add the user IDs and IP addresses for the Linux virtual servers you want to monitor.

   Example:

```
 FCONX    LINUXUSR A1  F 80  Trunc=80 Size=5 Line=4 Col=1 Alt=0

* * * Top of File * * *
LINUX01   190.10.10.115:8803
LINUX02   190.10.10.116:8803
LINUX03   190.10.10.117:8803
LINUX04   190.10.10.118:8803
LINUX05   190.10.10.119:8803
```

3. Save the file. From the XEDIT command line, issue:

```
====> file
```

Stay logged onto PERFSVM and continue with the next procedure.

(Optional) Steps for setting up the Web interface for Performance Toolkit for VM

Before you begin: You need to be logged on to PERFSVM. Later, you need to log onto TCPMAINT.

Perform these steps to set up the Web interface:

1. Edit FCONX $PROFILE A.
   a. From the command line, type this command and press the Enter key:

      ```
      xedit fconx $profile a
      ```

   b. Find the line with "MONCOLL WEBSERV". From the XEDIT command line, type this command and press the Enter key:

      ```
      ====> /moncoll webserv
      ```

   c. Replace the asterisk ("*") in column 1 with an "F" for the "MONCOLL WEBSERV" statement:

      ```
      FC MONCOLL WEBSERV ON TCPIP TCPIP 81
      ```

   d. In the prefix area for the FC MONCOLL WEBSERV statement, type "a", press the Enter key, and add this line:

      ```
      FC MONCOLL LINUXUSR ON
      ```

   e. Save the FCONX $PROFILE. From the XEDIT command line, type this command and press the Enter key:

      ```
      ====> file
      ```

2. Create an FCONRMT PASSFILE A.
   a. From the command line, type this command and press the Enter key:

      ```
      xedit fconrmt passfile a
      ```

   b. Add user IDs and passwords for users that you want to access performance data through the Web interface.

      Example:

      ```
       FCONRMT  PASSFILE A1  F 80  Trunc=80 Size=5 Line=5 Col=1 Alt=0
      DMSXMD573I Input mode:

      *USER-ID PASSWORD
      ******** ********
      DETRO    DETSPW
      MAINT    MAINTPW
      PERFGUY  CHCKPERF
      ```

   c. Press the Enter key twice to leave input mode.

d. Save the file. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

3. Log onto TCPMAINT.
4. Edit the PROFILE TCPIP. Issue this command:

```
xedit profile tcpip
```

5. Find the PORT statement. From the XEDIT command line, type this command and press the Enter key:

```
====> /port
```

6. Add port 81 to the PORT statement:

```
 PROFILE  TCPIP    A1  F 80  Trunc=80 Size=7 Line=5 Col=1 Alt=0

PORT
    21 FTPSERVE
    21 FTPSERV2
⋮
    81 TCP PERFSVM            ; Performance Toolkit for VM SERVER
```

7. Save the file. From the XEDIT command line, type this command and press the Enter key:

```
====> file
```

8. Because TCP/IP is running, dynamically add the port association. From the command line, issue:

```
netstat obey port 81 tcp perfsvm
```

You are done.

(Optional) Steps for testing your Web interface setup

Before you begin: You need be able to log off, then log back onto PERFSVM. You need to have a Web browser and know the host IP address for z/VM. You can get the address from Performance Toolkit for VM's basic screen initialization panel.

Example: http://9.60.27.206:00081/

Perform these steps to check your monitoring setup:

1. To re-initialize port 81:
   a. Log off PERFSVM.
   b. Log on PERFSVM.
2. Open your Web browser to the z/VM IP address and port number for Performance Toolkit for VM.
3. On the Web Server Login screen, type your user ID and password. The user ID and password must be one defined in the FCONRMT PASSFILE you defined previously (see step 2 on page 131).

4. Click the name of your z/VM host system.

   Result: You see:

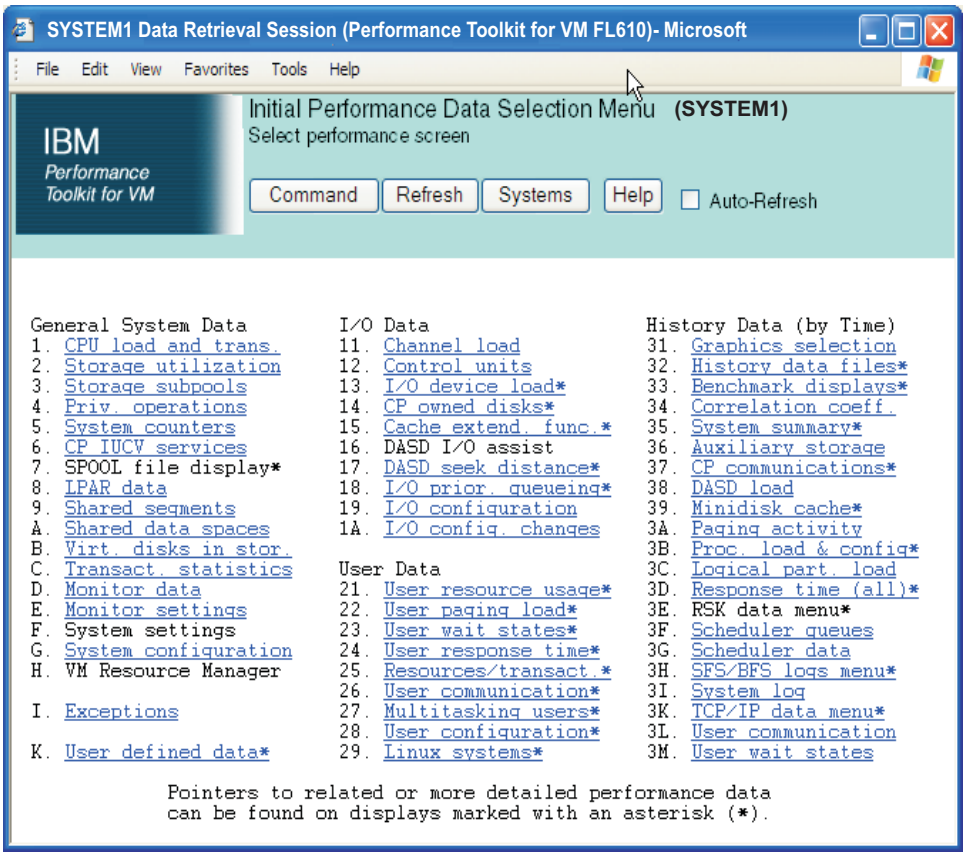

Figure 11. Initial Performance Data Selection Menu

5. Click "29. Linux Systems*" to see if the interface is displaying performance data from the Linux systems you configured.

   Tips:

   - A menu item is highlighted when data is available for that item.
   - Use the navigation buttons and links in the Web interface rather than your browser back and forward buttons.

You are done when you see the Linux performance data.

Using monitoring to analyze performance and capacity

This topic gives you a start on how to analyze performance and capacity problems. The topic does not cover performance and capacity in detail.

Related information:

- For more information on z/VM performance, see *z/VM: Performance*, SC24-6208.
- For detailed information about Performance Toolkit for VM screens, see *z/VM: Performance Toolkit Reference*, SC24-6210.

Steps for analyzing performance and capacity

Before you begin: You need to set up Performance Toolkit for VM and the Web interface for Performance Toolkit for VM. See "Configuring Performance Toolkit for VM" on page 124.

Perform these steps to analyze performance and capacity:

1. Open your Web browser to the z/VM host and port for Performance Toolkit for VM.
2. Base your next action on this table.

   Tip: For an explanation of the columns in a panel, click the column heading.

If you want to check for possible ...	Then check these screens ...	Notes
Storage constraints	For the whole system:	
	2. Storage utilization	Presents overall storage information for the z/VM system.
	For Linux guests:	
	21. User resource usage	
	22. User paging load	If user paging activity shows significant page migration rates below the 2 GB line (the ">2GB>" column) the system is constrained below the 2 GB line. In other words, the Control Program is thrashing on frames below 2 GB.
	29. Linux systems, then "LXMEM *userid*" where *userid* is the virtual machine user ID	If you look at the details of a particular virtual machine you see the storage details for that virtual machine: working set size, pages resident above and below 2G, and so forth. Check "Swap-in rate" and "Swap-out rate."
I/O constraints	11. Channel load	Busy channel paths on this screen can be an indication of I/O performance bottlenecks.
	13. I/O device load	This screen indicates I/O device throughput.

If you want to check for possible ...	Then check these screens ...	Notes
CPU constraints	For the whole system:	
	1. CPU load and trans.	Check the line: `PROC %CPU %CP %EMU %WT %SYS %SP %SIC %LOGLD` • "%LOGLD" is the best way to identify CPU bottlenecks. • "%CPU" through "%SP" are calculated on elapsed time and are a meaningful source for capacity planning.
	For Linux guests:	
	1. CPU load and trans.	Check "User Extremes:" (in the lower right of the panel) to identify heavy users or malfunctions of Linux guests (for example, looping).
	29. Linux systems 21. User resource usage	Check the Linux CPU utilization by entering "Linux *userid*" (*userid* is the virtual machine user ID). Linux tools like `top` report CPU consumption as if Linux owned the processors and do not consider that Linux is running under z/VM, so consumption figures differ with Performance Toolkit for VM.
Unwanted resource consumption	1. CPU load and trans.	• On Linux, use `ps -ef` to show the running tasks. Remove those tasks you do not need. • Install the on-demand timer patch, which disables the Linux wake-up function and allows z/VM to determine when a Linux guest is truly idle.

You are done.

Steps for using CP commands to improve performance

You can use certain CP commands to aid the performance of Linux guest operating systems. Remember that improving the performance of one machine can impair the performance of others.

Note
Do these activities only from a user ID for which QUICKDSP is set on. You can determine whether you have QUICKDSP on by issuing %CP QUERY QUICKDSP *userid*; look for the QUICKDSP setting in the response. To set QUICKDSP on, issue the %CP SET QUICKDSP *userid* ON command.

Before you begin: You need be able to log on as MAINT.

Perform these steps to use CP commands to enhance performance:

1. Log on as MAINT.
2. Base your action on the choices in the table:

If you want to ...	Then use this command ...

Control the fraction of system resources a virtual machine receives.	SET SHARE
	
	Example: By using these commands, you are establishing the fraction of system resources to which each virtual machine is entitled.
	`cp set share linux0 absolute 20%`
	`cp set share linux1 relative 100`
	`cp set share linux2 relative 300`
	`cp set share linux3 relative 100`
	
	Notes:
	1. A virtual machine receives its proportion of any scarce resource (CPUs, real storage, or paging I/O capability) according to its SHARE setting.
	2. The SET SHARE command can be used to set target minimum and maximum values (notice that LINUX0 has a target minimum of 20%—that means other virtual machines contend for the remaining 80% of resources).
	3. You can assign shares of system resources with the user directory SHARE statement so that when a user logs on, that virtual machine automatically has its share established.

Designate virtual machines that do not wait in the eligible list when they have work to do	SET QUICKDSP *userid*
	
	Example:
	`cp set quickdsp linux0 on`
	`USER LINUX0  :  QUICKDSP = ON`
	
	Note: With this setting, virtual machines are assigned an eligible list class of E0 and are added to the dispatch list immediately. QUICKDSP overrides the scheduler's usual resource assessment and fitting algorithms for the target virtual machine and the virtual machine is run without regard to its resource needs. When you use QUICKDSP, you take responsibility for resource allocation.

Guarantee that a certain minimum number of guest pages will always be resident	SET RESERVED **Example:** `cp set reserved linux01 50` means you are guaranteeing that the LINUX01 virtual machine has 50 pages always resident in real storage.

You know you are done when system performance improves.

Related information

For advanced information about performance and the z/VM scheduler, see

- *z/VM: Performance*, SC24-6208
- "VM Performance Resources" (http://www.vm.ibm.com/perf/)
- "The VM/ESA Scheduler Made Simple" (http://www.vm.ibm.com/devpages/bitner/presentations/vmsched.html)

Chapter 12. Servicing z/VM

This topic introduces IBM service concepts. After this introduction, turn to "Part 4. Service Procedure" in *z/VM: Guide for Automated Installation and Service*, GC24-6197, to install z/VM service.

z/VM service concepts

A component of z/VM, VMSES/E, helps you install z/VM and other VMSES/E-enabled products and apply service changes that correct or circumvent reported problems. VMSES/E handles both source code and binaries.

Service is the process of changing a particular release of a software product. There are a number of reasons for servicing a product, such as:

- Correcting a problem. Problems that you report to IBM are first entered into *problem management records (PMRs)*. If IBM determines that a PMR requires a fix to a product, IBM creates a record called an *authorized program analysis report (APAR)*. APARs provide a formal method of tracking problems for a specific version of a product. An APAR can also affect several releases of a product. IBM fixes these problems for a particular release through *program temporary fixes (PTFs)*. A PTF contains the code changes for a solution to a problem (APAR) on a particular release. Each release of a product has a unique set of PTFs, because the fixes may be different on each release. When IBM ships a new release of a product like z/VM, PTFs from the previous release are merged into the new release.
- Adding function. New functions can be delivered in a new release or version of a product, or as service. When new function is delivered as service, it is called a *small programming enhancement (SPE)*. SPEs are delivered and tracked the same way as problems. An APAR is assigned to the SPE, and it is delivered as a program temporary fix (PTF).

  IBM delivers service through:

  – *Recommended service upgrade (RSU)* tapes, which contain a collection of PTFs that IBM thinks are important enough that everyone should apply them. An RSU tape defines a *service level* for your product and is designed to prevent problems from occurring. (VMSES/E creates a software inventory and tracks the currency of the system through service levels.) This type of service is called preventive service.
  – *Corrective service (COR)* (delivered on tapes or sometimes electronically in electronic containers called *electronic envelopes*), which contains PTFs that you request for a specific problem. Sometimes you cannot wait for an RSU tape to correct a problem, so you can order a PTF through a COR tape or electronic envelope. This type of service is called corrective service because it is designed to correct a specific problem you encountered with the product.
  – *Expanded Service Options (ESOs)*, which are defined collections of PTFs delivered in VMSES/E corrective service format. ESO allows you to choose the starting and ending service levels.

    A product can have either an RSU or an ESO, but not both.

  Other reasons for changing a product are:
- Circumventing a problem. For expediency, IBM may provide a circumvention for a problem until a PTF can be developed. A circumvention is meant to be a

temporary solution to the problem, and it may be in the form of a procedural or software change. The method of delivery depends on the form of the solution, and it is determined by you and the IBM support center. Once an APAR number is established for the problem, you can use that number to track the fix and see when a PTF is available.

- Applying local service or modifications. *Local service* and *local modifications* are defined as any service or software change that is applied to your z/VM system that was not supplied by IBM through corrective service (COR) or a recommended service upgrade (RSU).

  When it is absolutely necessary to apply service from IBM before it is available on a COR tape, or when you need a local modification to tailor your system environment, you must apply the service locally. Local service includes updates supplied to you by other vendors or Licensed Products. Note that as subsequent PTFs are installed, both circumventions and local modifications may require rework.

  For more information on adding local modifications, see

  – "Apply a Local Modification" in *z/VM: Guide for Automated Installation and Service*, GC24-6197

  – "Procedures for Local Service and Modifications" in *z/VM: Service Guide*, GC24-6232.

In general, then:

- Someone reports a problem to IBM, which is entered as a problem management record (PMR).
- If it determines that a fix is needed, IBM creates an authorized program analysis report (APAR) for the code defect (bug).
- The solution for an APAR is a program temporary fix (PTF), which is like a code patch. The PTF might provide source updates in addition to new binaries.
- The PTF might be delivered as preventive service on a recommended service upgrade (RSU) tape. RSUs are periodic and intended for everyone to apply as a way of preventing problems.
- For problems you encounter that require immediate fixes, the PTF can be delivered on a corrective service (COR) tape or electronic envelope.
- Though PTFs usually contain fixes, sometimes IBM delivers new function through them, called small programming enhancements (SPEs).

Related Information

- If you need to install service, follow the instructions in "Part 4. Service Procedure" in *z/VM: Guide for Automated Installation and Service*, GC24-6197.
- For details on the service process, see "VM Service Concepts" in *z/VM: VMSES/E Introduction and Reference*, GC24-6243.

Appendix. Example of using FTP to install Linux from the hardware management console

If you do not have an external FTP server or do not want to create an external connection due to security concerns, you can install Linux by accessing a DVD or removable medium in the hardware management console (HMC) through FTP. You can link the hardware management console (HMC) DVD drive to your z/VM logical partition and ftp to the HMC DVD drive (using loopback FTP).

Notes:

1. This support is intended for customers who have no alternative, such as a LAN-based server, for serving the DVD contents for Linux installations. The elapsed time for installation using the HMC DVD drive can be an order of magnitude, or more, longer than the elapsed time for LAN-based alternatives.
2. If you need to avoid external connections, the z/VM FTP server must be connected through a guest LAN or VSWITCH to the virtual machine installing Linux. For more information, see *z/VM: TCP/IP Planning and Customization*.
3. To retrieve the initial boot files with FTP, you use the loopback address, but for the installation, when Linux asks for an FTP address to retrieve files, you use the home address of the z/VM TCP/IP server.
4. Because the directory containing the Linux boot files varies based on the Linux distribution and distribution level, the directory information used in the FTP GET commands in this topic is for example use only.

Linking the HMC removable media to your z/VM logical partition

Perform these steps to link the removable media in the HMC to the logical partition in which z/VM is running:

1. Open **CPC Images** on the HMC:
   a. From the Views area, double-click the **Task List**.
   b. From the Task List Work Area, open **CPC Recovery**.
   c. From the Views area, open **Groups**.
   d. From the Groups Work Area, open **CPC Images**.
2. Link the z/VM image to the HMC removable media:
   a. From the CPC Images Work Area, drag the z/VM image you want to link to and drop it on the **Access Removable Media** icon in the CPC Recovery Area. The Access Removable Media panel opens.
   b. On the Access Removable Media panel, click the appropriate option button to select the removable media that you want the image to access. Depending on which drives are available in the HMC, possible choices might include a DVD-RAM drive, USB flash memory drive, or diskette drive.

      Note: For supported USB flash memory drives, see *System z Hardware Management Console Operations Guide*.
   c. Click **OK** on the Access Removable Media panel.
   d. Click on **OK** in the Access Removable Media Task Confirmation panel. The Access Removable Media panel is displayed.

Important: Keep the Access Removable Media panel open until you finish the transfer (clicking **OK** on the panel closes access to the removable media). Only when you are done using the removable media should you click **OK** to end access.

FTPing to the HMC removable media

Before using FTP, you must grant authority in the FTP server to the user ID that accesses the HMC removable media. This user ID is the one you enter when the z/VM FTP client prompts `USER (identify yourself to the host):`. This example uses the user ID SIMONW.

You must also establish the HMC removable media as the default (initial) directory for the user ID accessing the removable media. The z/VM FTP server recognizes the fixed string `/../HMC:/` (case insensitive) to specify access to the removable media in the HMC. When there are multiple HMCs, the z/VM FTP server contacts the HMC that is linked to the LPAR using the Access Removable Media function described in "Linking the HMC removable media to your z/VM logical partition" on page 139.

These steps show how to grant authority and establish the initial directory. For the user ID accessing the HMC removable media, you must queue an `HMCAUTH YES` string followed by a `/../HMC:/` string to the program stack in the z/VM FTP server CHKIPADR exit.

Perform these steps to ftp to the HMC removable media through the z/VM FTP server:

1. For the user ID accessing the HMC removable media, grant authority to ftp to the HMC removable media through the z/VM FTP server. To grant authority, xedit the CHKIPADR SAMPEXEC file on the TCPMAINT 198 disk.
   a. Locate and remove the unconditional exit line (Exit 0).

```
/*--------------------------------------------------------------------*
/* Unconditional Exit - Simulate result of nonexistent exit.          *
/*--------------------------------------------------------------------*
/* To enable the SAMPLE STATEMENTS provided within this file (or to   *
/* enable the execution of your customized statements and logic),     *
/* DELETE the "Exit 0" statement that follows this comment block.     *
/*--------------------------------------------------------------------*
Exit 0    /* delete this line */
```

   b. Add the following case for the user ID that you use to access the removable media. For example, add the case for SIMONW:

```
When (Userid = 'SIMONW') Then
  Do
    Queue 'HMCAUTH YES'
    Queue '/../HMC:/'
  End
```

   c. Save the file as CHKIPADR EXEC.
   d. Restart the z/VM FTP server.
2. For your A-disk, access a disk large enough to hold the boot files (the 192 disk in the example) and then ftp to the loopback address as follows:

   Note: Parts in bold indicate commands you would issue.

```
access 192 a
DMSACC724I 192 replaces A (191)
Ready;
```

```
ftp 127.0.0.1
VM TCP/IP FTP Level 54

Connecting to 127...1, port 21
220-FTPSERVE IBM VM Level 54 at VM.DOMAIN.COM
220 Connection will close if idle for more than 5 minutes.
USER (identify yourself to the host):

simonw
>>>USER simonw
331 Password required for simonw.
Password:
>>>PASS <<<<<<<
230 User simonw logged in; working directory = /../HMC:/
Command:
```

3. Continue with the FTP transfer. Note the command `locsite fix 80`, which sets the VM file format to fixed length 80, the file format necessary for punching the binary files to the virtual machine reader.

```
bin
>>>TYPE i
200 Type set to I.
locsite fix 80
Command:
get /boot/s390x/vmrdr.ikr
>>>PORT 127,0,0,1,4,5
200 PORT command successful.
>>>RETR /boot/s390x/vmrdr.ikr
150 Opening BINARY mode data connection for vmrdr.ikr (1511884 bytes).
226 Transfer complete.
1511920 bytes transferred in 1.972 seconds. Transfer rate 766.69 Kbytes/sec.
Command:
get /boot/s390x/initrd suse.initrd
>>>PORT 127,0,0,1,4,7
200 PORT command successful.
>>>RETR /boot/s390x/initrd
150 Opening BINARY mode data connection for initrd (9862505 bytes).
8299680 bytes transferred.
226 Transfer complete.
9862560 bytes transferred in 12.366 seconds. Transfer rate 797.55 Kbytes/sec.
Command:
asc
>>>TYPE a
200 Type set to A.
Command:
get /boot/s390x/parmfile parm.file
>>>PORT 127,0,0,1,4,8
200 PORT command successful.
>>>RETR /boot/s390x/parmfile
150 Opening ASCII mode data connection for parmfile (38 bytes).
226 Transfer complete.
40 bytes transferred in 0.081 seconds. Transfer rate 0.49 Kbytes/sec.
Command:
quit
```

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 all 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, New York 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, New York 12601-5400
U.S.A.
Attention: Information Request

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

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 measurement 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.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain 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 may contain 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 Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

Adobe® and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

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

Java is a trademark of Sun Microsystems, Inc. in the United States, other countries, or both.

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

Glossary

For a list of z/VM terms and their definitions, see *z/VM: Glossary*.

The z/VM glossary is also available through the online z/VM HELP Facility. For example, to display the definition of the term "dedicated device", issue the following HELP command:

```
help glossary dedicated device
```

While you are in the glossary help file, you can do additional searches:

- To display the definition of a new term, type a new HELP command on the command line:

  ```
  help glossary newterm
  ```

  This command opens a new help file inside the previous help file. You can repeat this process many times. The status area in the lower right corner of the screen shows how many help files you have open. To close the current file, press the Quit key (PF3/F3). To exit from the HELP Facility, press the Return key (PF4/F4).
- To search for a word, phrase, or character string, type it on the command line and press the Clocate key (PF5/F5). To find other occurrences, press the key multiple times.

  The Clocate function searches from the current location to the end of the file. It does not wrap. To search the whole file, press the Top key (PF2/F2) to go to the top of the file before using Clocate.

Bibliography

See the following publications for additional information about z/VM. For abstracts of the z/VM publications, see *z/VM: General Information*.

Where to Get z/VM Information

z/VM product information is available from the following sources:

- z/VM Information Center at publib.boulder.ibm.com/infocenter/zvm/v6r1/index.jsp
- z/VM Internet Library at www.ibm.com/eserver/zseries/zvm/library/
- IBM Publications Center at www.elink.ibmlink.ibm.com/publications/servlet/pbi.wss
- *IBM Online Library: z/VM Collection on DVD*, SK5T-7054

z/VM Base Library

Overview

- *z/VM: General Information*, GC24-6193
- *z/VM: Glossary*, GC24-6195
- *z/VM: License Information*, GC24-6200

Installation, Migration, and Service

- *z/VM: Guide for Automated Installation and Service*, GC24-6197
- *z/VM: Migration Guide*, GC24-6201
- *z/VM: Service Guide*, GC24-6232
- *z/VM: VMSES/E Introduction and Reference*, GC24-6243

Planning and Administration

- *z/VM: CMS File Pool Planning, Administration, and Operation*, SC24-6167
- *z/VM: CMS Planning and Administration*, SC24-6171
- *z/VM: Connectivity*, SC24-6174
- *z/VM: CP Planning and Administration*, SC24-6178
- *z/VM: Getting Started with Linux on System z*, SC24-6194
- *z/VM: Group Control System*, SC24-6196
- *z/VM: I/O Configuration*, SC24-6198
- *z/VM: Running Guest Operating Systems*, SC24-6228
- *z/VM: Saved Segments Planning and Administration*, SC24-6229
- *z/VM: Secure Configuration Guide*, SC24-6230
- *z/VM: TCP/IP LDAP Administration Guide*, SC24-6236
- *z/VM: TCP/IP Planning and Customization*, SC24-6238
- *z/OS and z/VM: Hardware Configuration Manager User's Guide*, SC33-7989

Customization and Tuning

- *z/VM: CP Exit Customization*, SC24-6176
- *z/VM: Performance*, SC24-6208

Operation and Use

- *z/VM: CMS Commands and Utilities Reference*, SC24-6166
- *z/VM: CMS Pipelines Reference*, SC24-6169
- *z/VM: CMS Pipelines User's Guide*, SC24-6170
- *z/VM: CMS Primer*, SC24-6172
- *z/VM: CMS User's Guide*, SC24-6173
- *z/VM: CP Commands and Utilities Reference*, SC24-6175
- *z/VM: System Operation*, SC24-6233
- *z/VM: TCP/IP User's Guide*, SC24-6240
- *z/VM: Virtual Machine Operation*, SC24-6241
- *z/VM: XEDIT Commands and Macros Reference*, SC24-6244
- *z/VM: XEDIT User's Guide*, SC24-6245
- *CMS/TSO Pipelines: Author's Edition*, SL26-0018

Application Programming

- *z/VM: CMS Application Development Guide*, SC24-6162
- *z/VM: CMS Application Development Guide for Assembler*, SC24-6163
- *z/VM: CMS Application Multitasking*, SC24-6164
- *z/VM: CMS Callable Services Reference*, SC24-6165
- *z/VM: CMS Macros and Functions Reference*, SC24-6168

- *z/VM: CP Programming Services*, SC24-6179
- *z/VM: CPI Communications User's Guide*, SC24-6180
- *z/VM: Enterprise Systems Architecture/Extended Configuration Principles of Operation*, SC24-6192
- *z/VM: Language Environment User's Guide*, SC24-6199
- *z/VM: OpenExtensions Advanced Application Programming Tools*, SC24-6202
- *z/VM: OpenExtensions Callable Services Reference*, SC24-6203
- *z/VM: OpenExtensions Commands Reference*, SC24-6204
- *z/VM: OpenExtensions POSIX Conformance Document*, GC24-6205
- *z/VM: OpenExtensions User's Guide*, SC24-6206
- *z/VM: Program Management Binder for CMS*, SC24-6211
- *z/VM: Reusable Server Kernel Programmer's Guide and Reference*, SC24-6220
- *z/VM: REXX/VM Reference*, SC24-6221
- *z/VM: REXX/VM User's Guide*, SC24-6222
- *z/VM: Systems Management Application Programming*, SC24-6234
- *z/VM: TCP/IP Programmer's Reference*, SC24-6239
- *Common Programming Interface Communications Reference*, SC26-4399
- *Common Programming Interface Resource Recovery Reference*, SC31-6821
- *z/OS: IBM Tivoli Directory Server Plug-in Reference for z/OS*, SA76-0148
- *z/OS: Language Environment Concepts Guide*, SA22-7567
- *z/OS: Language Environment Debugging Guide*, GA22-7560
- *z/OS: Language Environment Programming Guide*, SA22-7561
- *z/OS: Language Environment Programming Reference*, SA22-7562
- *z/OS: Language Environment Run-Time Messages*, SA22-7566
- *z/OS: Language Environment Writing ILC Applications*, SA22-7563
- *z/OS MVS Program Management: Advanced Facilities*, SA22-7644
- *z/OS MVS Program Management: User's Guide and Reference*, SA22-7643

Diagnosis

- *z/VM: CMS and REXX/VM Messages and Codes*, GC24-6161
- *z/VM: CP Messages and Codes*, GC24-6177
- *z/VM: Diagnosis Guide*, GC24-6187
- *z/VM: Dump Viewing Facility*, GC24-6191
- *z/VM: Other Components Messages and Codes*, GC24-6207
- *z/VM: TCP/IP Diagnosis Guide*, GC24-6235
- *z/VM: TCP/IP Messages and Codes*, GC24-6237
- *z/VM: VM Dump Tool*, GC24-6242
- *z/OS and z/VM: Hardware Configuration Definition Messages*, SC33-7986

z/VM Facilities and Features

Data Facility Storage Management Subsystem for VM

- *z/VM: DFSMS/VM Customization*, SC24-6181
- *z/VM: DFSMS/VM Diagnosis Guide*, GC24-6182
- *z/VM: DFSMS/VM Messages and Codes*, GC24-6183
- *z/VM: DFSMS/VM Planning Guide*, SC24-6184
- *z/VM: DFSMS/VM Removable Media Services*, SC24-6185
- *z/VM: DFSMS/VM Storage Administration*, SC24-6186

Directory Maintenance Facility for z/VM

- *z/VM: Directory Maintenance Facility Commands Reference*, SC24-6188
- *z/VM: Directory Maintenance Facility Messages*, GC24-6189
- *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*, SC24-6190

Open Systems Adapter/Support Facility

- *System z10, System z9 and eServer zSeries: Open Systems Adapter-Express Customer's Guide and Reference*, SA22-7935
- *System z9 and eServer zSeries 890 and 990: Open Systems Adapter-Express Integrated Console Controller User's Guide*, SA22-7990
- *System z: Open Systems Adapter-Express Integrated Console Controller 3215 Support*, SA23-2247

Performance Toolkit for VM™

- *z/VM: Performance Toolkit Guide*, SC24-6209
- *z/VM: Performance Toolkit Reference*, SC24-6210

RACF® Security Server for z/VM

- *z/VM: RACF Security Server Auditor's Guide*, SC24-6212
- *z/VM: RACF Security Server Command Language Reference*, SC24-6213
- *z/VM: RACF Security Server Diagnosis Guide*, GC24-6214
- *z/VM: RACF Security Server General User's Guide*, SC24-6215
- *z/VM: RACF Security Server Macros and Interfaces*, SC24-6216
- *z/VM: RACF Security Server Messages and Codes*, GC24-6217
- *z/VM: RACF Security Server Security Administrator's Guide*, SC24-6218
- *z/VM: RACF Security Server System Programmer's Guide*, SC24-6219
- *z/VM: Security Server RACROUTE Macro Reference*, SC24-6231

Remote Spooling Communications Subsystem Networking for z/VM

- *z/VM: RSCS Networking Diagnosis*, GC24-6223
- *z/VM: RSCS Networking Exit Customization*, SC24-6224
- *z/VM: RSCS Networking Messages and Codes*, GC24-6225
- *z/VM: RSCS Networking Operation and Use*, SC24-6226
- *z/VM: RSCS Networking Planning and Configuration*, SC24-6227
- *Network Job Entry: Formats and Protocols*, SA22-7539

Prerequisite Products

Device Support Facilities

- *Device Support Facilities: User's Guide and Reference*, GC35-0033

Environmental Record Editing and Printing Program

- *Environmental Record Editing and Printing Program (EREP): Reference*, GC35-0152
- *Environmental Record Editing and Printing Program (EREP): User's Guide*, GC35-0151

Additional Publications

For white papers, IBM Redbooks publications, and other useful information about Linux on the mainframe, visit the z/VM resources for Linux on IBM System z Web site at:

http://www.vm.ibm.com/linux/

Publications you might be interested in are:

- *z/VM and Linux on IBM System z: The Virtualization Cookbook for SLES9*, SG24-6695
- *Security on z/VM*, SG24-7471

Index

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Program Number: 5741-A07

Printed in USA

SC24-6194-00