

# Chapter 5

---

## Virtual Partitions (vPars)

### Introduction

With Virtual Partitions (vPars) you can take almost any HP Integrity or HP 9000 server and turn it into many "virtual" computers. These virtual computers each run their own instance of HP-UX and associated applications. The virtual computers are isolated from one another at the software level. Software running on one Virtual Partition does not affect software running in any other Virtual Partition. In the Virtual Partitions you can run different patch levels of HP-UX, different applications, or any software you want and not affect other partitions.

There are differences between vPars running on HP 9000 and HP Integrity servers in that EFI, used on Integrity and described in Chapter 1, "Booting HP Integrity Servers," and ISL on HP 9000 are different. The other aspects of using vPars on the two platforms are very similar.

There are references in this chapter to bound and unbound CPUs. The latest version of vPars does not have a distinction of bound and unbound so you can disregard these references if you're using the newest version of vPars. Because there are many older vPars installations I've left the bound and unbound references in this chapter.

## About Virtual Partitions

There are some base requirements that must be met in order to run vPars on your system. At the time of this writing, the following minimum requirements must be met for each vPar on your system:

- Minimum of one CPU.
- Sufficient memory to run HP-UX and any other software that will be present in the vPar.
- A boot disk off which HP-UX can be booted. The example in this chapter uses a cell-based system. If you use the *internal* disks on cell-based systems for different vPars, the internal disks must be connected to different cell boards. If you boot off of a SAN, you must use separate Local Bus Adapters (LBAs.) On low-end systems that do not employ cell boards you only have to ensure that the boot devices are on separate LBAs.
- A console for managing the system. The console can be either physical or virtual. We cover the console later in this chapter.
- An HP Integrity server supported by HP-UX 11i Version 2 or later or an HP 9000 system supported by HP-UX 11i. Again, the example in this chapter will be an HP 9000 running Version 1 because this is the soft partition technology supported on HP 9000 since Integrity Virtual Machines are not available on HP 9000.

The system we use in most of the examples throughout this chapter is a small Integrity server that meets all the requirements in the previous list. You may also want to have additional disks and a separate LAN card in your vPars. I strongly recommend the LAN card so that you can establish Telnet or other sessions to your vPars rather than connecting to them only from the console. The LAN card is also required to perform backup and Ignite-UX-related work.

The vPars product is mature so you can confidently use it in production environments, keeping in mind that full software isolation is employed in vPars but not hardware isolation. If you need the additional confidence of hardware isolation, then you want to use nPartitions, which are covered in "Node Partitions (nPartitions) and Management Processor Overview" chapter.

## Virtual Partitions Background

HP-UX Virtual Partitions (vPars) allow you to run multiple instances of HP-UX on the same HP Integrity or 9000 Server. From a hardware perspective, a vPar consists of CPU, memory, and I/O that is a subset of the overall hardware on the computer. From a software perspective, a vPar consists of the HP-UX 11i Operating Environment and all application-related software to successfully run your workload. The following figure shows a conceptual diagram of the way in which HP computer-system resources can be allocated to support multiple vPars.

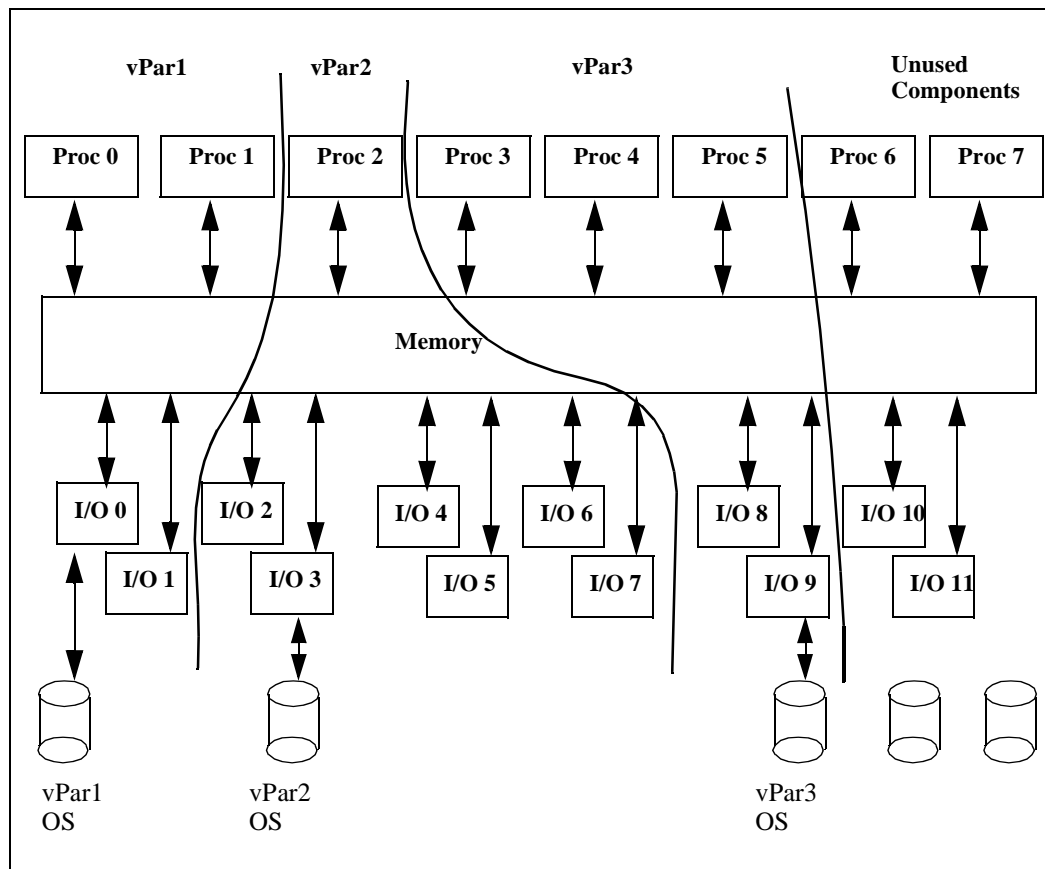


Figure 5-1 Example of HP System Resource Allocation with vPars

The components of which your HP server is comprised can be allocated in a variety of ways. You can see that the eight-way system shown in the figure has a different number of processors, different amount of memory, and different number of I/O cards allocated to each vPar. The unused components can be added to any of the vPars or be the basis for yet another vPar. In addition, components can be moved from one vPar to another (with some restrictions described later in this chapter). Cell boards are not depicted in the figure. There are some nuances related to working with vPars on cell-based systems that will be described in an example in this chapter.

## Uses of Virtual Partitions

I have worked on many vPars installations that have a variety of uses for vPars. The following is a sampling of the reasons to use vPars:

- |                              |   |
|------------------------------|---|
| Increased System Utilization | Many servers are underutilized. With vPars, you can devote a subset of system resources to each vPar. With each vPar running its own instance of HP-UX 11i and associated applications, you get higher overall system utilization.  |
| Quick Deployment             | You can deploy a new environment quickly without procuring an entire new system.  |
| Flexibility                  | Many applications have resource needs that change. With vPars, you can devote fewer system components when application needs are low and additional resources when an application needs them. An increased end-of-the-month workload, for example, can be given more system resources to complete faster. |
| Server Consolidation         | Running multiple instances of HP-UX 11i and their associated applications on one HP server reduces the overall number of servers required. Web servers that had run on different servers can now run in different vPars on the same computer.   |

Application Isolation      HP vPars are fully software-isolated from one another. A software failure in one vPar does not affect other vPars.

#### Mixed Production, Test, and Development

Production and testing can take place on the same server with vPars. When testing is complete, the test vPar can become the production vPar. Similarly, development usually takes place on a separate system. With the software isolation of vPars; however; development can take place on the same system with other applications.

These are just a sampling of the uses I've seen for vPars. Many others will emerge as vPars become widely used and systems experts implement them in more computing environments.

## Loading the Software

The "Installing HP-UX," chapter covers loading HP-UX 11i in detail. If you haven't before loaded HP-UX 11i, this chapter helps you complete the task of loading 11i on all of the disks that you will use for your vPars. Based on the previous discussion of disks, we might load HP-UX 11i on the internal disk at path *0/0/1/1.2.0* and the device at path *0/8/0/0.8.0.5.0.0.0*. Chapter 3 will walk you through the process of selecting a target device on which to load HP-UX 11i as well as the process of loading 11i. The following is a bullet list of steps you need to perform on every disk that will act as a vPar boot device:

1. Install the HP-UX 11i *Operating Environment*.
2. Set system parameters at the time of first boot after loading HP-UX 11i with **set\_parms**.
3. Download and install select patches on your system (at the time of this writing there are many patches required to support vPars.)
4. Install vPars software.
5. Configure vPars.

6. Install additional software.

You would typically load software in the order just shown: Install the *Operating Environment*; boot your system and use **set\_parms**; load patches; install vPars software; configure vPars; and then install and configure all other software.

I cover installing Virtual Partitions software in this section. I assume that you already have HP-UX 11i installed on your system or know how to do so. If you have not yet installed HP-UX 11i, see Chapter 3, which covers installing HP-UX 11i.

Keep in mind that HP-UX must be loaded for each Virtual Partition you want to run. If, for example, you want to run two Virtual Partitions, as we do in our examples in this book, HP-UX 11i needs to be loaded for both Virtual Partitions. The procedure covered for loading HP-UX 11i needs to be performed for every Virtual Partition you want to run. HP-UX 11i can be loaded from media, such as your HP-UX 11i distribution on a CD-ROM or from an Ignite/UX server. You can use any method to load HP-UX 11i and the Virtual Partitions software for every Virtual Partition you want to run.

The following figure shows an example of the software components that appear for vPars software.

```
File View Options Actions Help
Press CTRL-K for keyboard help.
Source: meta2:/tmp/vpar/vpar.tar
Target: meta2:/
Only software contained in the parent bundle is shown.
Only software compatible with the target is available for selection.

Subproducts or Filesets:T1335AC.VirtualPartition 0 of 3 selected

Marked? Name Revision Information
[ ] (go up)
[ ] VPAR-KRN -> A.03.01.05 Virtual Partition Kernel
[ ] VPAR-MON2 -> A.03.01.05 Virtual Partition Monito
[ ] VPAR-RUN -> A.03.01.05 Virtual Partition User S
```

Figure 5-2 Example of Loading vPars Software

The figure shows the components of which vPars software is comprised.

All the vPars software must be loaded on every HP-UX 11i volume that will be used on your vPars server. The loading of this software takes place for every HP-UX 11i instance that you want to run simultaneously on your vPars server. After loading this software, you can run the following **swlist** command to see the filesets:

```
[metal] / # swlist -l fileset | grep Vir
# VirtualPartition A.03.01.05 HP-UX Virtual Partitions Functionality
  VirtualPartition.VPAR-KRN A.03.01.05 Virtual Partition Kernel Files
  VirtualPartition.VPAR-MON A.03.01.05 Virtual Partition Monitor
  VirtualPartition.VPAR-RUN A.03.01.05 Virtual Partition User Space Commands
[metal] / #
```

Your revision number of vPars software will be newer than mine, which was just released at the time of this writing.

There are two ways to load the HP-UX 11i operating system and vPars software on all the volumes used for vPars. The first, which is the method used throughout this book, is to load HP-UX 11i and vPars software on all vPars volumes prior to creating Virtual Partitions. The second is to load only the volume of the first vPar with all software, create as many vPars as you want, and then use **vparboot -p vp\_name -I ignite\_kernel** to boot and load HP-UX 11i on the other disks. Using Ignite/UX you have to specify the full path of the kernel, as shown in this command:

```
vparboot -p <name> -I /opt/ignite/boot/Rel_B.11.11/WINSTALL
```

In this chapter, I first load HP-UX 11i and vPars software on all disks before creating vPars.

A lot of software has been loaded as a result of loading the vPars software. The **/sbin** directory has in it the *vpar* commands we'll use in upcoming sections. The following is a long listing of the *vpar* commands in **/sbin**:

```
[metal] / # ll /sbin/vpar*
-r-xr-xr-x 1 bin bin 120624 Apr 8 15:49 /sbin/vparboot
-r-xr-xr-x 1 bin bin 148224 Apr 8 15:49 /sbin/vparcreate
-r-xr-xr-x 1 bin bin 87800 Apr 8 16:06 /sbin/vpard
-r-xr-xr-x 1 bin bin 54976 Apr 8 16:06 /sbin/vpardump
-r-xr-xr-x 1 bin bin 30240 Apr 8 16:06 /sbin/vparextract
-r-xr-xr-x 1 bin bin 132000 Apr 8 15:49 /sbin/vparmodify
-r-xr-xr-x 1 bin bin 47712 Apr 8 16:06 /sbin/vparreloc
-r-xr-xr-x 1 bin bin 114488 Apr 8 15:49 /sbin/vparremove
-r-xr-xr-x 1 bin bin 119288 Apr 8 15:49 /sbin/vparreset
-r-xr-xr-x 1 bin bin 147880 Apr 8 15:49 /sbin/vparstatus
-r-xr-xr-x 1 bin bin 25728 Apr 8 16:06 /sbin/vparutil
[metal] / #
```

These are the commands that you use to create, view, modify, and work with vPars in general.

There are several files in **/stand** related to the vPars kernel. The following listing shows some of them:

```
[metal] / # ll /stand/vp*
-rw----- 1 root    root          20520 Sep 17 11:56 /stand/vpdb
-r-xr-xr-x 1 bin     bin           1168728 Apr  8 15:44 /stand/vpmon
-rw----- 1 root    root          18350080 Sep 17 10:47 /stand/vpmon.dmp
[metal] / #
```

**vpmon** is loaded at the time of system startup and is the basis for running vPars. **vpdb** is the vPars database that contains all information related to all the vPars running on your system. This file is automatically synchronized by the vPars monitor to ensure that all vPars have the same information about all vPars on your system. **vpmon.dmp** is the vPars dump file.

There are several startup-related files, including those shown below, which are covered in more detail in Chapter 8, "System Startup and Shutdown."

```
[metal] / # ll /etc/rc.config.d/vpar*
-r--r--r-- 1 bin     bin           291 Oct 27 2001 /etc/rc.config.d/vpard
-r--r--r-- 1 bin     bin           553 Oct 29 2003 /etc/rc.config.d/vparhb
-r--r--r-- 1 bin     bin           1246 Oct 27 2001 /etc/rc.config.d/vparinit

[metal] / # ll /sbin/init.d/vpar*
-r-xr-xr-x 1 bin     bin           793 Oct 27 2001 /sbin/init.d/vpard
-r-xr-xr-x 1 bin     bin           922 Oct 27 2001 /sbin/init.d/vparhb
-r-xr-xr-x 1 bin     bin           7808 Aug 13 2003 /sbin/init.d/vparinit
[metal] / #
```

Of particular interest is **vparhp**, which is the *heartbeat* daemon related to keeping **vpdb** synchronized on all of your vPars.

Very important to your work related to vPars are the online man pages. The following listing shows the man pages loaded on my system at the time of this writing:

```
[metal] / # man -k vpar
vparboot(1M)      - boot a virtual partition
vparcreate(1M)   - create a virtual partition
vpardump(1M)     - manage monitor dump files
vparextract(1M)  - extract memory images from a running virtual partition system
vparmodify(1M)   - rename a virtual partition or modify the attributes of a
                  virtual partition
vparreloc(1M)    - relocate the load address of a vmunix file, determine if a vmunix
```

```
file is relocatable, or promote the scope of symbols in a
relocatable vmunix file
vparremove(1M) - remove a virtual partition
vparreset(1M) - reset a virtual partition
vparresources(5) - description of virtual partition resources and their
requirements
vparstatus(1M) - display information about one or more virtual partitions
vpartition(5) - display information about the Virtual Partition Command
Line Interface
vparutil(1M) - get and set SCSI parameters for SCSI controllers from a
virtual partition
[metal] / #
```

You may have to run the **catman** command to create cat files for these manual pages.

At this point, we have HP-UX 11i and the Virtual Partitions software loaded on the system.

The remainder of this chapter covers numerous vPars topics, including creating, booting, and modifying vPars.

With both HP-UX 11i and the Virtual Partitions software on our disk, we can begin the process of creating partitions. Our goal is to have a system that looks like what's shown in the following figure.

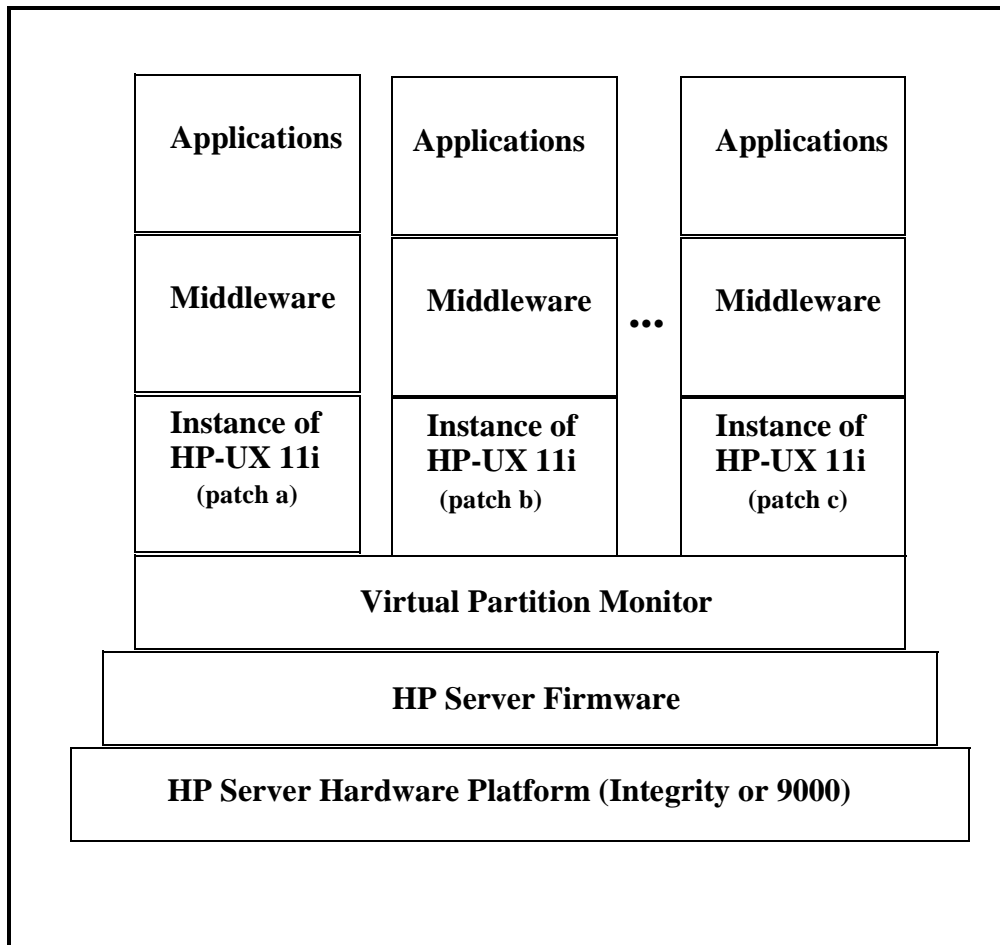


Figure 5-3 Virtual Partitions Software Stack

There are many components in this figure. We already have many of the components in this diagram on our system. Starting from the bottom, we have the hardware, firmware, Virtual Partition Monitor, and HP-UX 11i installed on two different disks. There are two HP-UX 11i instances shown in the left-most two stacks of the figure. These are the operating systems that we have already loaded.

The two HP-UX 11i instances can't run simultaneously on a system because we have not yet created our Virtual Partitions. Without Virtual Partitions created, we can boot HP-UX off of one *or* the other of these disks, but

we can't run both. Let's now create our Virtual Partitions so that we can have two instances of HP-UX 11i running simultaneously. After Virtual Partitions have been created, you can proceed to load the middleware and applications shown on top of the previous figure.

## Virtual Partitions Command Summary

There are several commands used to create and work with Virtual Partitions. A table and a tear-out card in my vPars book, *HP-UX Virtual Partitions*, provide an overview of many commonly used Virtual Partitions related commands. The following table is an abbreviated version of the command summary:

**Table 5-1** Virtual Partition Commands

Command	Description
<p><i>ISL</i>&gt;</p> <p>Initial System Load prompt.</p>	<p>Virtual Partitions Monitor is loaded from <i>ISL</i>&gt; with:</p> <pre>ISL&gt; hpux /stand/vpmon</pre> <p><i>MON</i>&gt;</p> <p>To load Virtual Partitions directly from <i>ISL</i>&gt;, use:</p> <pre>ISL&gt; hpux /stand/vpmon vparload -p vPar_name</pre>
<p><i>MON</i>&gt;</p> <p>Virtual Partitions Monitor prompt. (Also see <b>vparload</b> command.)</p>	<p>This is loaded from <i>ISL</i> with:</p> <pre>ISL&gt; hpux /stand/vpmon</pre> <p><i>MON</i>&gt;</p> <p>To load an alternate database from <i>ISL</i>, use:</p> <pre>ISL&gt; hpux /stand/vpmon -D db_file</pre> <p>To load one vPar from <i>MON</i>, use:</p> <pre>MON&gt; vparload vPar_name</pre> <p>Many other commands can be issued from <i>MON</i>. Type <b>help</b> or <b>?</b> to list. (Commands include: <b>scan</b>, <b>vparinfo</b>, <b>ls</b>, <b>log</b>, <b>getauto</b>, <b>lifs</b>, <b>cbuf</b>, <b>cat</b>.)</p>
<p><b>vparload</b></p> <p>Load Virtual Partitions from <i>MON</i>&gt; prompt only.</p>	<p>To boot a Virtual Partition from <i>MON</i>&gt;:</p> <pre>MON&gt; vparload -p vPar_name</pre>

Command	Description
<b>vparboot</b>  Boot a Virtual Partition from the command line only.	To boot a Virtual Partition from the command line: <pre># vparboot -p vPar_name</pre>
<b>vparcreate</b>  Create a Virtual Partition.	To create a Virtual Partition with three processors ( <i>num</i> ) total, two bound ( <i>min</i> ), 2048MB RAM, all components on 0/0, boot disk at 0/0/1/1.2.0, with a kernel of <i>/stand/vmunix</i> , autoboot on, and console at 0/0/4/0:  <pre># vparcreate -p vPar_name -a cpu::3 -a cpu:::2:4 -a mem::2048 -a io:0/0 -a io:0/0/1/1.2.0:boot -b /stand/vmunix -B auto</pre>
<b>vparmodify</b>  Modify the attributes of a Virtual Partition.	To add processor at path <i>IO9</i> (adds this proc to those already assigned): <pre># vparmodify -p vPar_name -a cpu:109</pre>
<b>vparremove</b>  Delete a Virtual Partition.	To delete a Virtual Partition in the currently running database: <pre># vparremove -p vPar_name</pre>
<b>vparreset</b>  Reset a Virtual Partition.	To reset a Virtual Partition without TOC (t), hard (h), bypassing display of PIM data (q), or forcing (f):  <pre># vparreset -p vPar_name</pre>
vparresources(5) man page  Provides description of Virtual Partitions and their resources.	This is a manual page that describes Virtual Partition resources in general and how resources are specified in other commands, such as <b>vparmodify</b> .
<b>vparstatus</b>  Display the status of Virtual Partitions.	To display the status of a Virtual Partition in verbose mode: <pre># vparstatus -v -p vPar_name</pre>

Command	Description
vpartition man page  Display information about the Virtual Partition Command Line Interface.	Provides the following brief description of Virtual Partitions commands: <b>vparboot</b> Boot (start) a virtual partition. <b>vparcreate</b> Create a new virtual partition. <b>vparmodify</b> Modify an existing virtual partition. <b>vparremove</b> Remove (delete) an existing virtual partition. <b>vparreset</b> Simulate a TOC or hard reset to a virtual partition. <b>vparstatus</b> Display virtual partition and available resources information.
Specify CPU Resources by:	Number of bound and unbound CPUs: <i>cpu::num</i> CPU hardware path(s): <i>cpu:path</i> Minimum and maximum number: <i>cpu::[min][:[max]]</i>
Specify Memory by:	Size <i>mem::size</i> Base and range: <i>mem::base:range</i> combination of both above.
Specify I/O:	Use path: <i>io:path[:attr1[,attr2[...]]]</i> (see man page <b>vparresources</b> for details).
To add resources use: (This adds component relative to what already exists if running <b>vparmodify</b> .)	<i>-a cpu:path</i> <i>-a cpu::num</i> (can be done with vPar running) <i>[-a cpu::num] [-a cpu::[min]:[max]] [-a cpu:path] (:: is <b>vparcreate</b> only)</i> <i>-a io:path[:attr1[,attr2[...]]]</i> <i>-a mem::size</i> <i>-a mem::base:range</i>
To delete resources use (This deletes component relative to what already exists if running <b>vparmodify</b> .)	<i>-d cpu:path</i> <i>-d cpu::num</i> (can be done with vPar running) <i>-d io:path[:attr1[,attr2[...]]]</i> <i>-d mem::size</i> <i>-d mem::base:range</i>
To modify resources use: (This modifies to absolute number rather than relative.)	<i>-m cpu::num</i> (can be done with vPar running) <i>-m cpu::[min][:[max]]</i> <i>-m io:path[:attr1[,attr2[...]]]</i> <i>-m mem::size</i>

Command	Description
vPars <b>setboot</b> Options: <i>-a</i> <i>-b</i> <i>-p</i> <i>-s</i> no options	Changes the alternate boot path of the Virtual Partition. Sets the autoboot attribute of the Virtual Partition. Changes the primary boot path of the Virtual Partition. No affect. Displays information about boot attributes.  To set Autoboot <i>on</i> : # <b>setboot -b on</b>
vPars States: <i>load</i>  <i>boot</i>  <i>up</i> <i>shut</i> <i>down</i> <i>crash</i> <i>hung</i>	The kernel image of a Virtual Partition is being loaded into memory. This is done by the Virtual Partition monitor. The Virtual Partition is in the process of booting. The kernel image has been successfully loaded by the Virtual Partition monitor. The Virtual Partition has been successfully booted and is running. The Virtual Partition is in the process of shutting down. The Virtual Partition is not running and is down. The Virtual Partition has experienced a panic and is crashing. The Virtual Partition is not responding and is hung.

We use some of the commands shown in this table in the upcoming section on creating virtual partitions.

Let's now move on to creating virtual partitions.

## Steps to Create Virtual Partitions

This section covers the steps to create Virtual Partitions. I performed these steps while working with some of my clients on their vPars environments. This list should serve as a framework for working with vPars. You may chose not to perform some of the steps and to add others. It is only a framework for getting vPars working on your system.

In our upcoming examples to create our Virtual Partitions, we execute the steps shown in the following figure.

- 1) Load HP-UX 11i onto the disks on which you want to run a Virtual Partition\* (media or Ignite/UX server.)
- 2) Load Virtual Partitions software onto the disk(s) on which you want to run a Virtual Partition.
- 3) Gather information on system components and hardware paths using **ioscan**, **dmesg**, and other commands.
- 4) List components of which Virtual Partitions will be comprised such as:
  - name
  - CPUs
  - memory
  - others
- 5) Create first Virtual Partition with **vparcreate**.
- 6) Boot first Virtual Partition with **vparload** at *MON*> prompt. Use **vparstatus -v** to see running vPar and **vparstatus -A** to see available components.
- 7) Create second Virtual Partition with **vparcreate** (can also be done before booting any vPars.)
- 8) Boot second Virtual Partition with **vparboot** from the first vPar and monitor it booting with **vparstatus**. After it has booted, view remaining available components with **vparstatus -A**.
- 9) Modify Virtual Partition(s) as required with **vparmodify** and view modifications with **vparstatus -v** and **vparstatus -A**. Modifications can be any type, such as adding CPUs, changing attributes, and so on.

Other tasks and comments:

- Many other tasks can be performed. Commands **vparremove** and **vparreset** were not used in steps above.

- A typical list of components of which a vPar would be comprised looks like the following:

```

name          vpar1
processors    min of 1 (bound) max of 3 (1 bound 2 unbound) with num equal to 1
memory        1024 MB
LAN           0/0/0/0 (not specified explicitly)
boot disk     0/0/1/1.2.0
kernel        /stand/vmunix
autoboot      off (manual)

```

\* HP-UX 11i must be loaded on volume before or after Virtual Partition is created. If HP-UX 11i is loaded after vPar is created, then `vparboot -p vp_name -I ignite_kernel` is used to load 11i.

Figure 5-4 Steps to Create Virtual Partitions

## 1) Load HP-UX 11i

HP-UX 11i must be loaded on the volumes that will be used to host all vPars. The method you use to install 11i, whether media, Ignite-UX, or some other technique, are all acceptable provided that HP-UX 11i is present on all the disks. HP-UX 11i must be present on the first disk before you begin the vPar creation. You can create vPars on other disks before HP-UX 11i is loaded on them and then use **vparboot -p *vp\_name* -I *ignite\_kernel*** to boot and load HP-UX 11i on the other disks. In this chapter, I first load HP-UX 11i on all disks before creating vPars. Loading HP-UX and vPars software was previously covered in this chapter. In the upcoming example, two virtual partitions are created on two different internal disks on an rp8420.

HP-UX and vPars software can be loaded on internal disks, which will be covered in the upcoming example, or on a Storage Area Network (SAN.) The following listing shows the internal disk as the boot path for vPar *metal* and a boot disk on the SAN for vPar *meta2*:

```
[metal] / # vparstatus -v -p metal
vparstatus: Warning: Virtual partition monitor not running, Requested resources shown.
[Virtual Partition Details]
Name:      metal
State:     N/A
Attributes: Dynamic, Autoboot
Kernel Path: /stand/vmunix
Boot Opts:

[CPU Details]
Min/Max:  1/<default>
Bound by User [Path]:
Bound by Monitor [Path]: <no path>
Unbound [Path]:  <no path>
                  <no path>
                  <no path>
                  <no path>
                  <no path>
                  <no path>
                  <no path>
                  <no path>
                  <no path>
                  <no path>
                  <no path>
                  <no path>

[IO Details]
0.0.0.2.0.6.0.0.0.0.0 BOOT          <- internal SCSI boot disk

[Memory Details]
Specified [Base /Range]:
          (bytes) (MB)
Total Memory (MB): 24576
```