A Dissertation

ON

Managing Power Servers and Virtualization Features Through a Single Communication Channel

Submitted for partial fulfillment of the award of

MASTER OF TECHNOLOGY

Degree in

SOFTWARE ENGINEERING

By

CHIRAG MITTAL
(2K11/SWE/05)

Under the guidance of

Ms. DIVYASHIKHA SETHIA

2013

DEPARTMENT OF COMPUTER ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
DELHI-110042

DECLARATION

I hereby declare that the thesis entitled "**Managing Power Servers and Virtualization Features Through a Single Communication Channel"** which is being submitted to the Delhi Technological University, in partial fulfillment of the requirement for the award of degree of Master of Technology in Software Engineering is an authentic work carried out by me.

Chirag Mittal
(2K11/SWE/05)
Department of Computer Engineering
Delhi Technological University
Delhi – 110042

CERTIFICATE

Delhi Technological University
(Government of Delhi NCR)
Bawana Road, New Delhi-42

This is to certify that the thesis entitled "**Managing Power Servers and Virtualization Features Through a Single Communication Channel**" done by Chirag Mittal (Roll Number: 2K11/SWE/05) for the partial fulfillment of the requirements for the award of degree of Master of Technology Degree in Software Engineering in the Department of Computer Engineering, Delhi Technological University, New Delhi is an authentic work carried out by him under my guidance.

Ms. Divyashikha Sethia
Assistant Professor
Department of Computer Engineering
Delhi Technological University
Bawana Road, Delhi- 110042

ACKNOWLEDGEMENT

This dissertation would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost, my utmost gratitude to my guide **Ms. Divyashikha Sethia** and my mentor at IBM, India **Mr. Hariganesh Muralidharan (HMC Development, IBM)** and **Ms. Rashmi Narasimhan (Architect-Service Processor Development, IBM)** for providing excellent guidance, encouragement and inspiration throughout the project work.

I am heartily thankful to **Mr. Ajay K. Mahajan (Power Firmware Architect, IBM) and Ms. Kiranmai Kuchibhotla (Manager – Power HMC Systems, IBM)** for the constant support and timely requisite resources for the successful completion of the project.

Last but not least, I would like to thank my **family** and **friends** specially **Divya Mittal** and **Garima Mittal** who have been a source of encouragement and inspiration throughout the duration of the project.

Chirag Mittal

Acronym

AIX - Advanced Interactive Executive

AMS - Active Memory Sharing

DLPAR - Dynamic logical partitioning

FSP - Flexible Service Processor

HMC – Hardware Management Console

IBM – International Business Machine

LAN – Local Area Network

LPAR - Logical Partition

MAC - Mandatory Access Control

PHYP - Power Hypervisor

PLIC - Platform Licensed Internal Code

PowerVM – Power Virtual Machine

pSeries – Power Series

SAN – Storage Area Network

SCSI – Small Computer System Interface

SSL – Secure Socket Layer

TCE - Translation Control Entry

VM - Virtual Machine

VMM-Virtual Machine Monitor

Abstract

Virtual machine technology, or virtualization, is gaining momentum in the information technology community. While virtual machines are not a new concept, recent advances in hardware and software technology have brought virtualization to the forefront of IT management. Stability, cost savings, and manageability are among the reasons for the recent rise of virtualization. IBM PowerVM allows business units to consolidate multiple workloads onto fewer systems, increasing server utilization, and reducing cost. Power Hypervisor (PHYP) provides virtualization capabilities to IBM PowerVM. Hardware Management Console (HMC) manages virtualization features on Power Servers. HMC communication for configuration information happens via the Flexible Service Processor (FSP) to PHYP over a dedicated management network.

In Power Server, FSP is the conduit for Virtualization traffic from HMC. We have reduced the load of FSP by redirecting the PHYP targeted commands to logical partition. In this work, we have proposed an architecture in which a logical partition running Linux will be an interface for all PHYP targeted commands. This will reduce the latency of managing virtualization features on Power Servers. We have created logical partition running Linux on Power Server and implemented Network Server (NETS) (written in C++) on logical partition. Network Server establishes a Secure Socket Layer (SSL) connection with HMC, accepts the command issued by HMC and routes the commands to PHYP. This architecture reduces traffic over FSP henceforth improves the efficiency. Commands which are directed for server diagnosis, recovery and status information will be issued by HMC to FSP and commands targeted to PHYP for virtualization features will be issued by HMC to logical partition running Linux.

Contents

Declaration *i*

Certificate *ii*

Acknowledgement *iii*

Acronym *iv*

Abstract *v*

List of Figure(s) *viii*

1 Introduction

1.1 Overview 1

1.2 Thesis Objective and Scope 3

1.3 Thesis Outline 3

2 Literature Survey 4

3 IBM PowerVM Virtualization 7

3.1 Introduction of PowerVM virtualization 7

3.2 PowerVM Architecture 8

3.3 Components of POWERVM 9

3.3.1 Hardware Management Console 9

3.3.2 Flexible Service Processor 11

3.3.3 Logical Partition 12

3.3.4 Power Hypervisor 13

3.4 PowerVM Features 18

3.4.1 Micro Partitioning 18

3.4.2 Active Memory Sharing 20

3.4.3 Virtual Input Output 27

4 Implementation and Performance Analysis 32

4.1 Concept 32

4.2 Architecture 33

4.3 Impact of Architecture on Performance 38

5 Conclusion and Future Work 40

References 42

List of Figures

Figure 1.1 Communication between HMC and PHYP through FSP

Figure 2.1: Software Virtual Machine

Figure 2.2: Hardware Virtual Machine

Figure 2.3: virtual OS/Container Virtual Machine

Figure 3.1: IBM PowerVM Architecture

Figure 3.2: HMC Types

Figure 3.3: Represents AIX, i5/OS, and Linux logical partitions installed on IBM Systems

Figure 3.4: POWER Hypervisor: Abstraction of physical server

Figure 3.5: POWER Hypervisor Architecture

Figure 3.6: Power Physical Shared Processor Pool and micro-partition

Figure 3.7: Dedicated Memory Model

Figure 3.8: Shared Memory Model

Figure 3.9: Active Memory Sharing Architecture

Figure 3.10: Logical Memory Mapping

Figure 3.11: Processing an I/O request from Virtual Machine

Figure 3.12: Modern Device Split Device Virtualization

Figure 3.13: VIOS operation in PowerVM

Figure 4.1: HMC interacting with Linux partition and service processor

Figure 4.2: Two communication channel by HMC for interaction with Power Server

Figure 4.3: Reduction in execution times of operations

Snapshot 4.1: Execution over command prompt

Snapshot 4.2: Command prompt showing HMC and logical partition connection state is connected

Chapter 1 Introduction

1.1 Overview

Virtualization is a technology which is used to improve IT throughput, system utilization and total cost. Virtualization layer hides complexity and heterogeneity of the underlying hardware and software so that several virtual machines can run on top of it. It combines physical resources into shared pools to receive virtual resources, which can be assigned to virtual machines running on top of virtualization layer.

Virtualization can provide the following benefits:

- Virtualization reduces operation and system management costs and provides efficient access and management of resources.
- Virtualization enables a single server to perform as multiple virtual servers.
- Virtualization provides dynamic sharing of resources pools to increase the use of existing resources.

IBM PowerVM technology provides virtualization of IBM power servers. PHYP which comes under proprietary of IBM supports virtualization technologies, logical partitioning and dynamic resource movement among operating systems which are running on logical partitions. Dynamic resource movement provides Processor, memory and I/O to move dynamically among logical partitions depending on their workloads. Hypervisor also supports many advance features which includes virtual

I/O and high speed communication between partitions using virtual LAN. It also enforces partition security and can provide virtual LAN channels between logical partitions, reducing the need for physical Ethernet adapters and releasing I/O adapter slots. The Hardware Management Console (HMC) is used to configure and control one or more managed systems. HMC is used to create and manage logical partitions and activate Capacity Upgrade on Demand. As shown in Figure 1.1 HMC communicates with the PHYP through FSP, which is used to configure, diagnostic, initialization of Power Server.

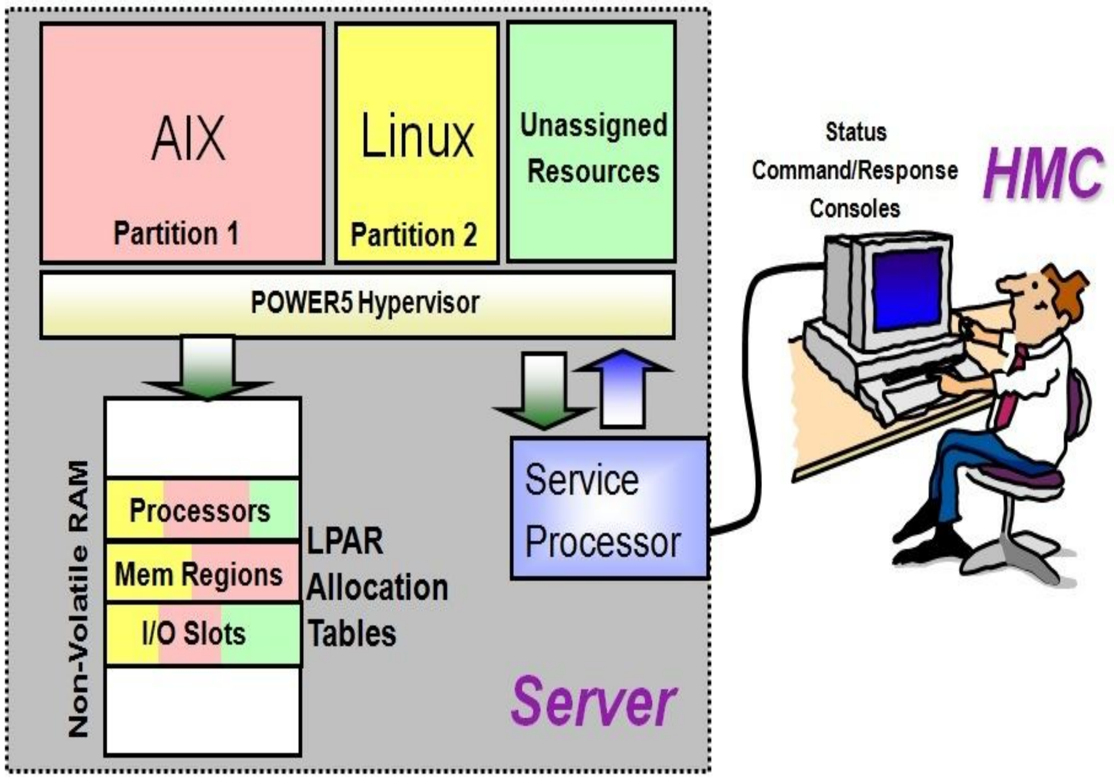


Figure 1.1: Communication between HMC and PHYP through FSP

1.2 Thesis Objective and Scope

HMC is used to manage the IBM Power Servers. HMC is used to create logical partitions, management of the partitions, service related functions and advanced virtualization features.

The platform management is handled through a FSP. FSP is the conduit for Virtualization traffic from HMC. The goal of this thesis is to create a logical partition running Linux that will be interface for all PHYP targeted commands (as mentioned in section 4.2). This will remove FSP from the critical path of the virtualization and reduce the latency of deploying and managing partitions by removing the FSP path and directly interacting with PHYP.

1.3 Thesis Outline

Thesis consists of following chapters:

Chapter 2 explains the studies carried out in literature. **Chapter 3** explains about PowerVM architecture, components and features. **Chapter 4** explains proposed concept and architecture by which HMC can interact with PHYP through logical partition running Linux, and impact on performance of new architecture compare to existing. At last we summarize the thesis with Conclusion and Future work in **Chapter 5.**

Chapter 2 Literature Survey

- **Virtual Machines and Implementation**

Jeff Daniels in his paper '**Server Virtualization Architecture and Implementation**' [7] explains virtual machine and the types in which they can be implemented. He explained that virtual machine can be implemented in three ways:

1. Software Virtual machine, in which hypervisor sits in between host operating system and guest operating system. Hypervisor manages the interaction between host machine and virtual machine.

Figure 2.1 explains software virtual machine architecture.

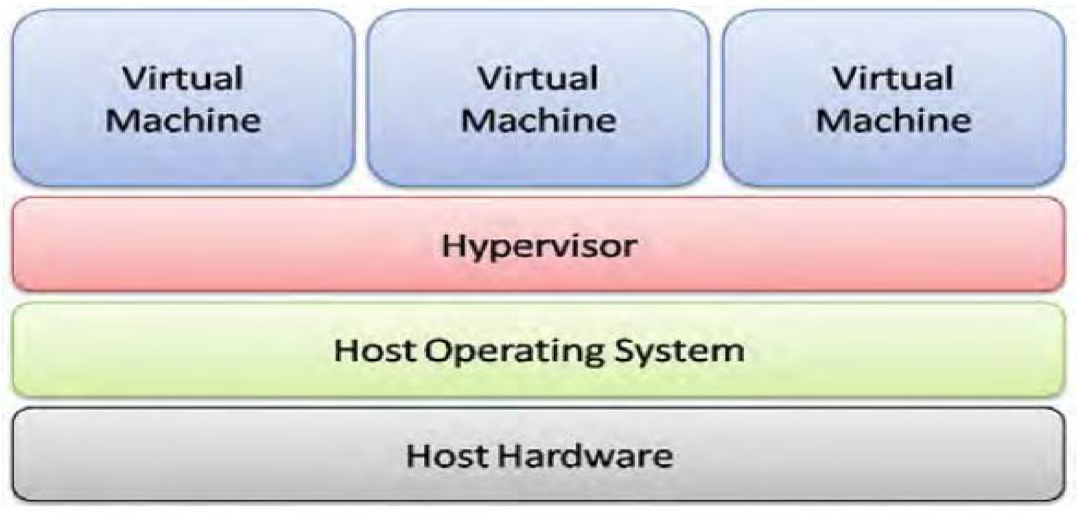


Figure 2.1: Software Virtual Machine [7]

2. Hardware virtual machines (Figure 2.2), virtualization technology (hypervisor) sits directly on host hardware. Virtual machines runs on top of hypervisor.

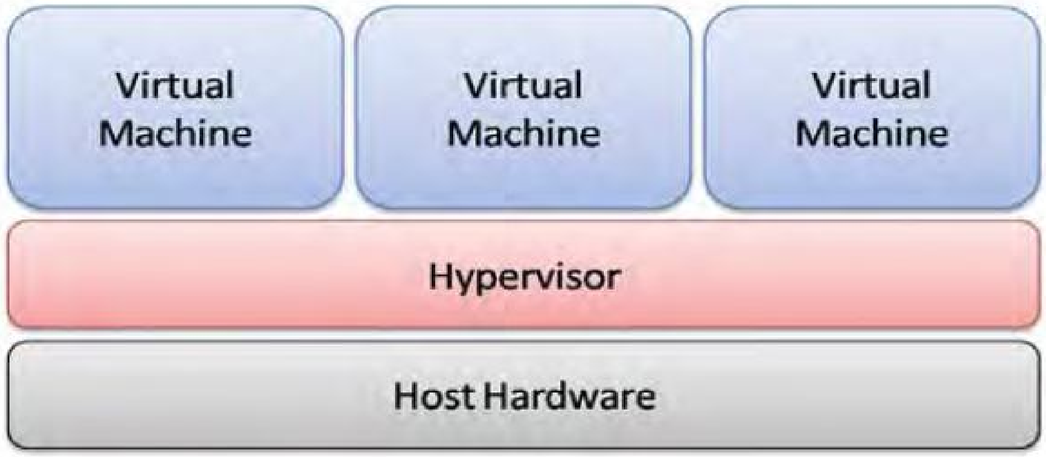


Figure 2.2: Hardware Virtual Machine [7]

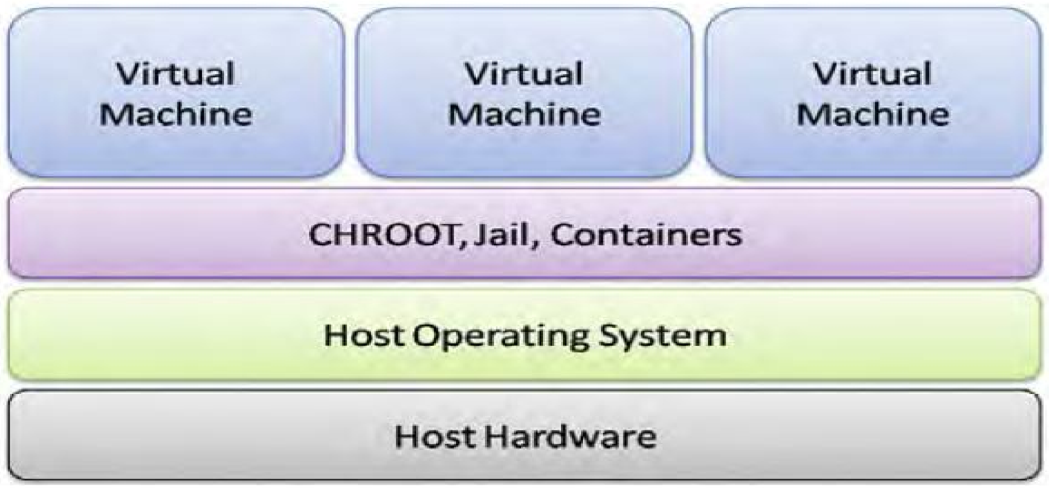


Figure 2.3: virtual OS/Container Virtual Machine [7]

3. Virtual OS/containers (Figure 2.3), in which the host operating system is partitioned into containers or zones.

- **IBM Power Platform Architecture**

W. J. Armstrong et al., 2005 in their paper '**Advanced virtualization capabilities of POWER5 system'** [1] proposed about IBM Power Platform architecture. They explained that POWER processor combined with PHYP provides IBM with leading edge virtualization and partitioning support in the latest power servers. Power Server provides paravirtualization or cooperative partitioning with AIX, i5/OS, and Linux operating systems, which results in an excellent combination of virtualization capabilities with minimal overhead. Virtualization features provides consolidation of multiple operating system on a single platform, dynamic reconfiguration of partitions resources, efficient resource utilization through recovery of ideal processing cycles.

- **PHYP and Resource Management**

Enriquillo Valdez et al., 2007 in their paper '**Retrofitting the IBM POWER Hypervisor to Support Mandatory Access Control'**[1] described the design and architecture of PHYP and provide a workload protection mechanism for PHYP to mediate resource assignment by mandatory access control (MAC) on LPARs. PHYP consists of non-blocking interrupt driven layer, called Platform Licensed Internal Code (PLIC), and a multitasking kernel, called Dispatchable PHYP.

Chapter 3 IBM PowerVM Virtualization

3.1 Introduction of PowerVM Virtualization

PowerVM is a server virtualization technology developed by IBM. It allows business units to consolidate multiple application workloads like Database, web servers etc. onto fewer systems, increasing server utilization and reducing cost. PowerVM provides a secure and scalable server virtualization environment for IBM AIX/ i-series and Linux based applications built upon the IBM Power "P" series server platforms optimized for virtualization. Key capabilities of PowerVM technology on IBM Power series servers are to enable optimal hosting of Business applications in Data Centers. PowerVM technology optimizes server utilization and I/O resource sharing to provide better use of IT assets, and provides business applications to be hosted at lesser cost ownership.

PowerVM optimizes the performance of the business application by improving its response time by dynamic allocation of resources as needed by the application.

The Power Hypervisor sits above the layer of hardware resources to act as a layer of abstraction to hide the physical hardware resources from the application workloads seating above, while controlling the distribution of the hardware resources to these workloads.

3.2 PowerVM Architecture

According to the existing structure of IBM Power Servers HMC communicates with managed systems via FSP through a secure communication channel. In the process HMC first creates a connection with FSP by sending series of configuration information and validating the response, after wards FSP sends command to PHYP which are received from HMC and targeted for PHYP. Figure 3.1 shows architecture of IBM PowerVM.

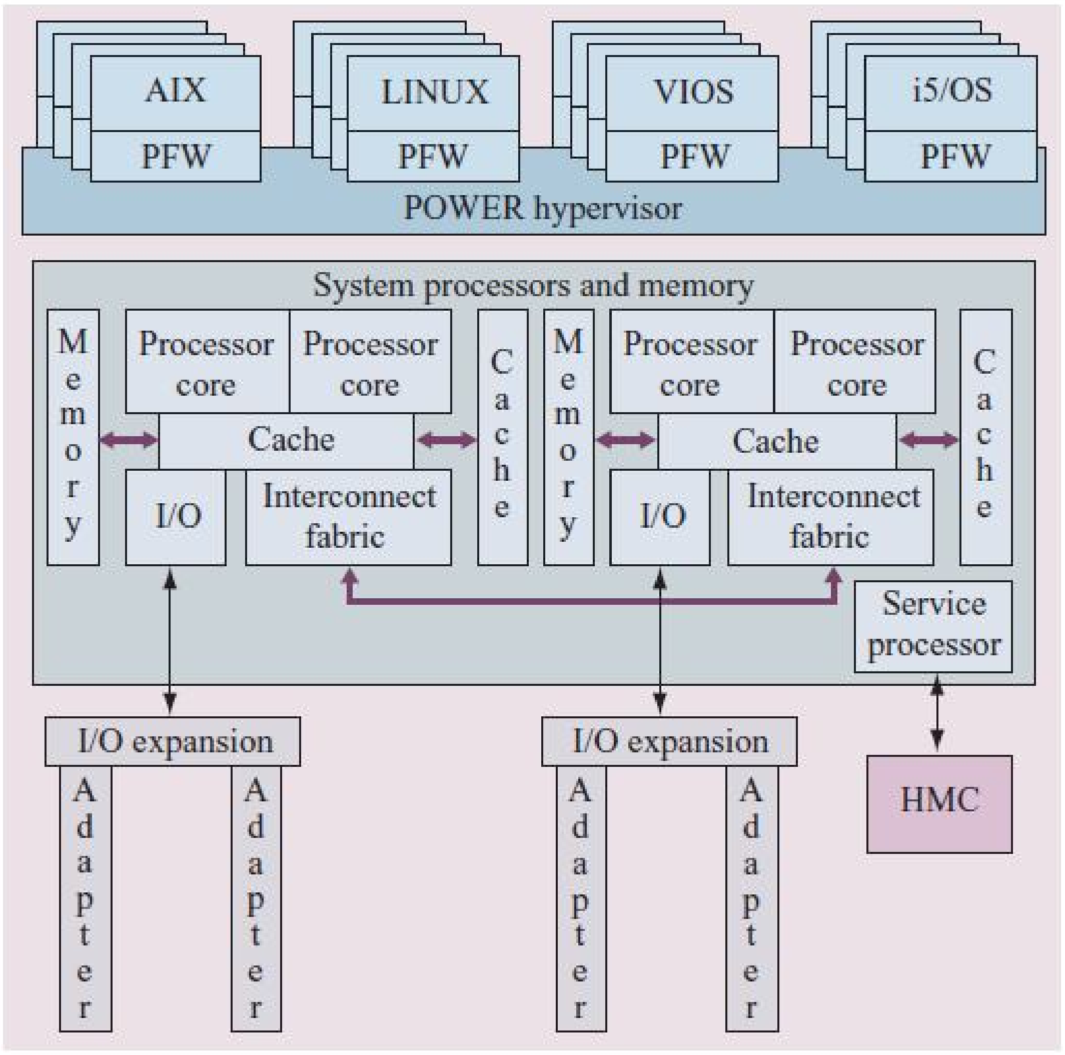


Figure 3.1 IBM PowerVM Architecture [12]

The hypervisor layer is responsible for validating OS requests to use shared platform resources, thus ensuring the integrity and isolation of these partitioned systems. Operational management for the platform is performed via the HMC. The HMC manages platform resources via messages to hypervisor.

3.3 Components of POWERVM

3.3.1 Hardware Management Console

Nowadays, among the overall cost IT organizations are spending, the cost of managing the IT infrastructure has become the largest and fastest-growing component. To address this cost, IBM Systems Director platform management family comprises of HMC [4]. It actually assists the IT professionals with the tools and techniques which they require to have a better coordination and management for all of their virtual and physical resources present in the Server. Though, it also enhances complexity of the system by increasing the number of managed virtual resources significantly.

HMC creates the initial configuration definition, controls boot and termination of the various partitions, and provides console support. It is the control point for dynamic reconfiguration of the resources of the partition, and deferred and maintenance of Power Servers.

Functions of HMC

Using HMC, a system admin can do multiple functions such as, logical partitioning functions, service functions, and different system management functions by using either the web-browser-based user interface or the command-line interface (CLI).

We can hereby summarize the basic functionalities of HMC as follows:

- Management of virtualization features.
- Power management for systems as ON and OFF.
- Creation and maintenance of logical partitions (LPAR's) in a managed system.
- Perform dynamic LPAR (DLPAR) operation.
- Display of managed system resources and status.

HMC types

There are some models of HMC that were available for pSeries. These models can be upgraded to run the HMC code and manage power systems. Figure 3.2 shows there are two types of HMC models available:

- Desktop HMC.
- Rack mounted HMC.

Desktop HMC

The supported desktop models are 7310-C03 models. On the desktop, we can connect a keyboard and mouse to the standard keyboard, mouse PS/2 style connectors, or USB ports. We cannot connect any other devices to the HMC.

Rack Mounted HMC

The IBM 7310–CR2 Rack-mounted HMC provides a dedicated workstation designed for configuration and management of logical partitions and Capacity on Demand (CoD) for POWER5, or later, processor-based servers.

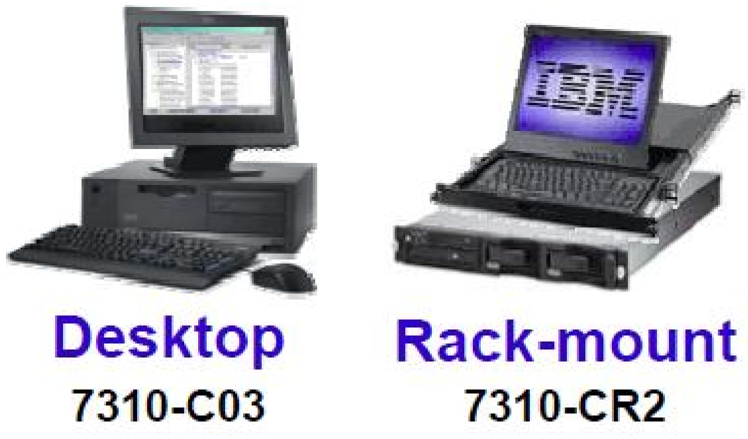


Figure 3.2 HMC Types [4]

3.3.2 Flexible Service Processor (FSP)

HMC communicates to power servers through FSP. FSP runs on its own power boundary and continually monitors hardware attributes and the environmental conditions within the power server.

FSP is responsible for loading the PHYP code from Flash into system memory and starting the system processors. FSP directly communicates only with the Real Control Panel, PHYP, or the HMC.

FSP firmware provides:-

- Diagnostics
- Initialization
- Configuration
- Run-time error detection, and correction.
- Power and thermal management
- Serviceability (Error logging and Dump management, Concurrent Maintenance)

We are reducing the load of FSP by directing PHYP related commands to some other logical partition running Linux.

3.3.3 LPAR

Logical partitioning was introduced with the POWER4 processor-based product line and the AIX Version 5.1 operating system. LPAR technology provides capability to divide a pseries system into separate logical systems. Each logical partition can run an individual operating system. Hardware resources such as processors, memory and I/O components can be assigned to logical partitions by virtualization. So LPAR's and virtualization increases utilization of system resources and add a new level of configuration possibilities.

These are some characteristics of LPAR:-

- Each LPAR is independent.
- Each LPAR runs its own operating system.
- Each partition owns a defined amount of real storage.
- Strictly no storage shared across partitions.
- Separation of logical partitions is considered as perfect as having each logical partition on a separate server.
- CPUs may be dedicated to a logical partition or may be shared by multiple partitions.
- When shared, each LPAR is assigned a number of logical processors (up to the maximum number of physical processors).
- I/O channels may be dedicated to a partition or may be shared by multiple partitions.
- Each LPAR has its own architecture mode.

Figure 3.3 shows three logical partitions which runs AIX, i5/OS and linux operating system respectively.

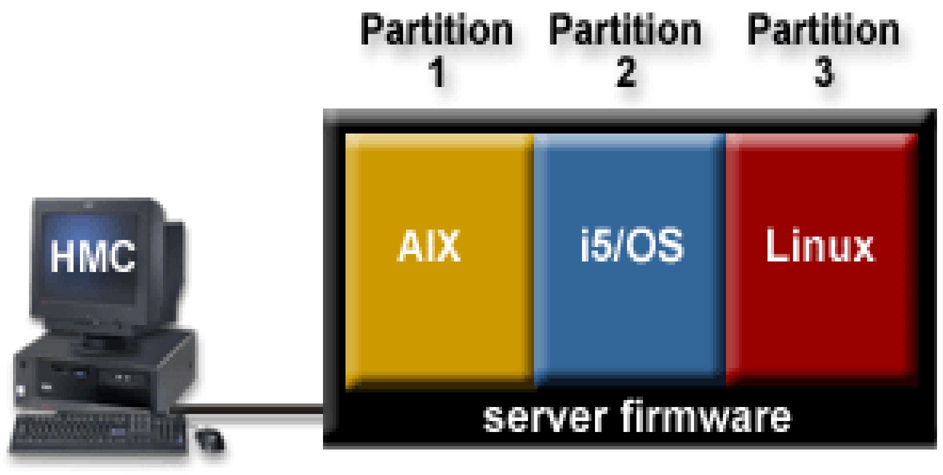


Figure 3.3: Three logical partitions running AIX, i5/OS, and Linux on IBM Power Systems

Dynamic logical partitioning increased flexibility by allowing system resources such as memory, I/O and processor to be added and deleted from the logical partitions dynamically. This provides optimal use of system resources by reconfiguring resource among logical partitions according to requirements.

3.3.4 Power Hypervisor (PHYP)

Hypervisor is termed as virtual machine monitor (VMM), used to run multiple virtual machines simultaneously on a single host. This is an application of virtualization due to which we can increase server utilization and decrease power, space and time requirements. Virtual machines are identified by distribution of the tasks and services in a workload. Using Hypervisor, we can conventionally separate individual virtual machine in order to enable the sharing of resources on platform.

Hypervisor can be classified into two types based on underlying operating environment:

- TYPE 1 Hyper visor
- TYPE 2 Hypervisor

Type 1 Hypervisor runs directly on the hardware. There is no underlying mediator operating system. The hypervisor controls access to the hardware, and an individual virtual machine sits on top of the hypervisor. So effectively there is only one software layer between the hardware and the virtual machine.

Type 2 Hypervisor runs within a conventional operating system environment. The underlying operating system simultaneously runs other applications – in addition to the hypervisor application without affecting the working of hypervisor. Virtual machines run on top of the hypervisor layer which in turn signifies that there are two layers of software between hardware and each virtual machine– the hypervisor and the host operating system.

PHYP lays the foundation of IBM PowerVM. The basic functionalities of PHYP comprise of Micro-Partitioning, dedicated – processor partitions, virtual processor, virtual SCSI adapters, and virtual Ethernet adapters. It is also responsible for providing guaranteed isolation to virtual machines, these isolations being termed as LPAR. By using PHYP, we can prevent programs running in one LPAR from affecting other programs running in other LPAR. Also, it separates LPAR memories, and allows exclusive physical device assignments to LPAR. This exclusive physical device assignment to LPAR is done by administrators through a stand-alone HMC.

POWER Hypervisors (PHYP) performs mainly the following functions:

- Provides security layer between logical partitions and enforces partition integrity.
- Provides a layer of abstraction between the physical hardware resources and the logical partitions which are using them. As well as, it controls the dispatch of virtual processors to physical processors, and simultaneously saves and restores all processor state information during virtual processor context switching.
- It controls management structure and services for partitions along with hardware I/O Interrupts.

Figure 3.4 shows how physical server hardware abstraction among partitions is provided by PHYP.

As we know, the exclusive physical device assignment to LPARs is done by administrators through a stand-alone HMC. HMC is used to run the management application, is a dedicated PC. It provides the interface for configuring and managing the platform. The HMC communication for configuration information happens via FSP to PHYP over a dedicated management network. Here, the communication is done using the SSL protocol and password protection. As explained in section 3.2.2 FSP being an independent subsystem maintains platform configuration information and performs system diagnostics.

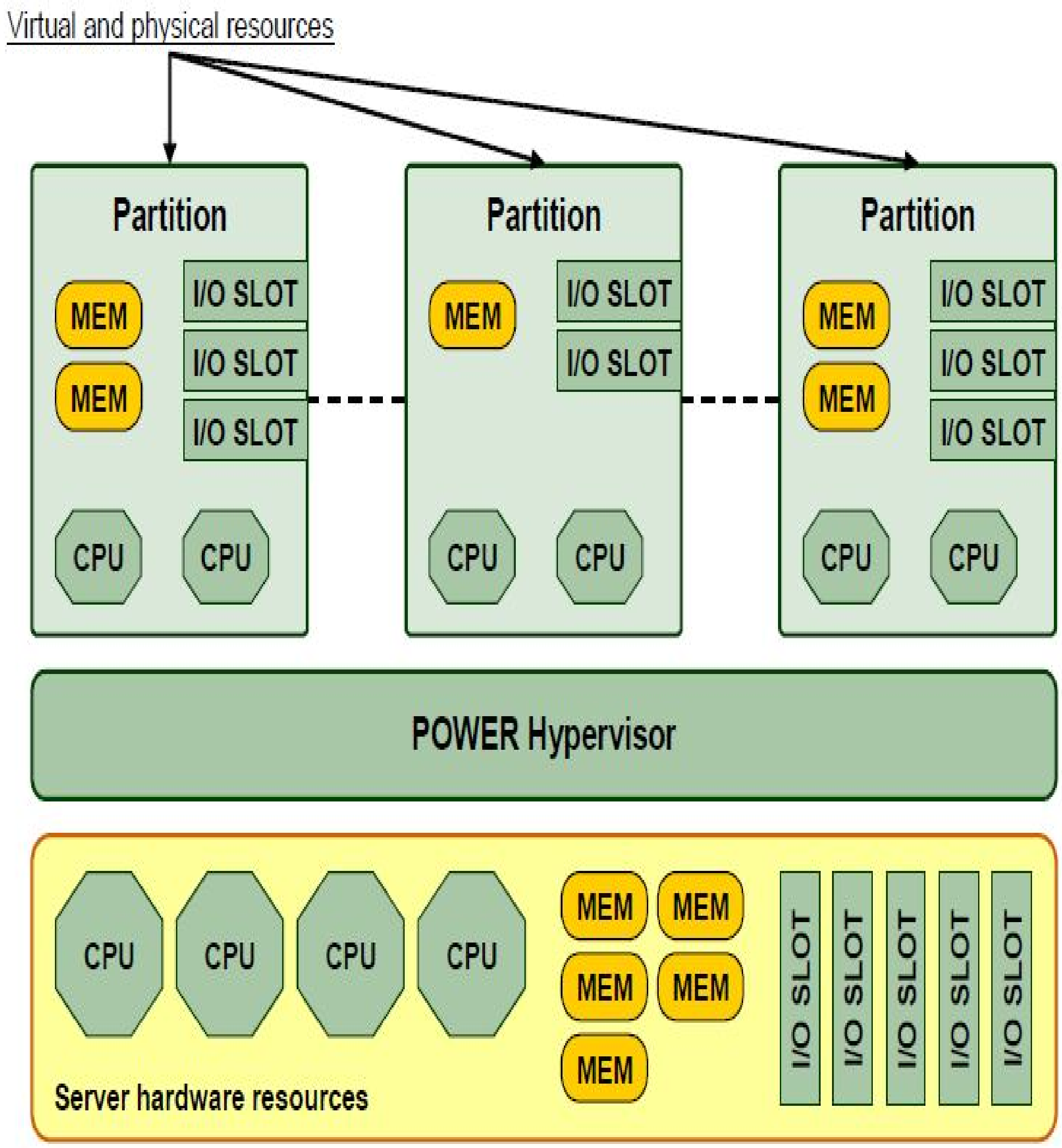
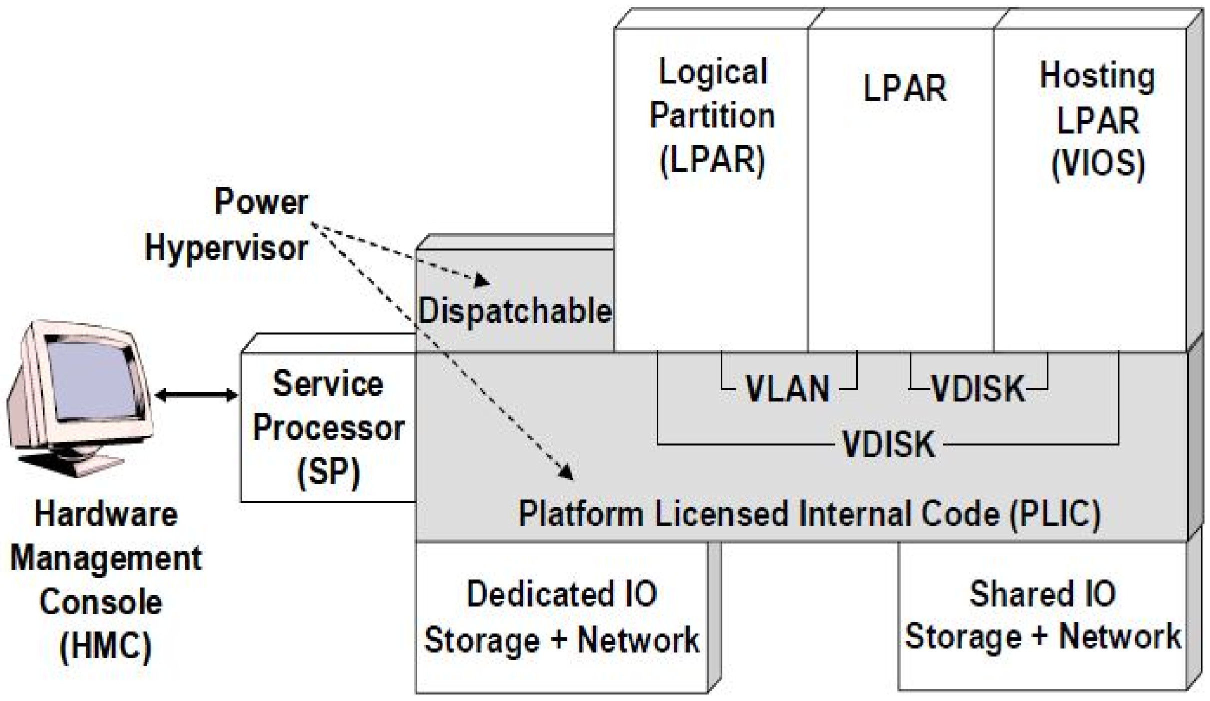


Figure 3.4 POWER Hypervisor: Abstraction of physical server hardware [3]

PHYP is divided into two layers: a non-blocking interrupt driven layer, called Platform Licensed Internal Code (PLIC), and a multitasking kernel, called Dispatchable PHYP (Figure 3.5) [1]. For virtualization, time critical operations are required which are performed by PLIC. For example, it enforces the partitioned environment. PLIC also maintains the hardware page tables for translating an LPAR memory address into a physical address. Along with it, it validates an LPAR's access to its hardware page table entry. Due to this, it prevents an LPAR from accessing the memory of another LPAR which in turn saves time and effort. Similarly, it maintains an IO memory map unit in the form of a Translation Control Entry (TCE) table, which is used to translate addresses generated by IO devices to physical memory which is then assigned to LPARs.



Logical Partition (LPAR)
LPAR
Hosting LPAR (VIOS)
Power Hypervisor
Dispatchable
Service Processor (SP)
VLAN
VDISK
VDISK
Platform Licensed Internal Code (PLIC)
Hardware Management Console (HMC)
Dedicated IO Storage + Network
Shared IO Storage + Network

Figure 3.5 POWER Hypervisor Architecture [1]

Dispatchable PHYP executes as a hidden LPAR and is used to provide non-critical platform services. One of its primary duties is to process messages from the HMC and FSP. The HMC provides Dispatchable PHYP with LPAR configuration data. Dispatchable PHYP is responsible for processing and maintaining configuration data which can be in the form of messages, on the platform. It is involved in every configuration update whether it is static or dynamic. Even in the processing of Dynamic LPAR operations, dispatchable PHYP is used where resources are added or removed on running LPAR. Dispatchable PHYP is also used for directing the startup and terminations of LPAR.

3.4 POWERVM Features

3.4.1 Micro Partitioning

Micro-Partitioning is the technique used to distribute the processing capacity of one or more processor among logical partitions. So micro partitioning provides significantly increase in overall uses of hardware processor in system. The micro-partition is provided with a processor entitlement—the processor capacity guaranteed to it by the PHYP.

Many technologies associated with micro-partitioning:

- Physical shared-processor pool
- Micro-partitions
- Shared dedicated capacity
- Multiple Shared-Processor Pools

Micro Partition

IBM power Server provides virtualization of physical processor which is implemented by PHYP. PHYP hides the physical processor from the logical partitions and presents as a set of virtual processors within micro partition.

The operating systems running on LPAR dispatches executable tasks to these virtual processor. A micro-partition can have a processor entitlement from a minimum of 0.1 of a processor up to the total processor capacity in the system.

Shared Processor Pools

Shared Processor combines processor resources to be combine in a pool which can be shared among LPAR. It increases efficient utilization of processors within the system.

Figure 3.6 shows relationship between micro-partition, virtual processors and shared-processor pool.

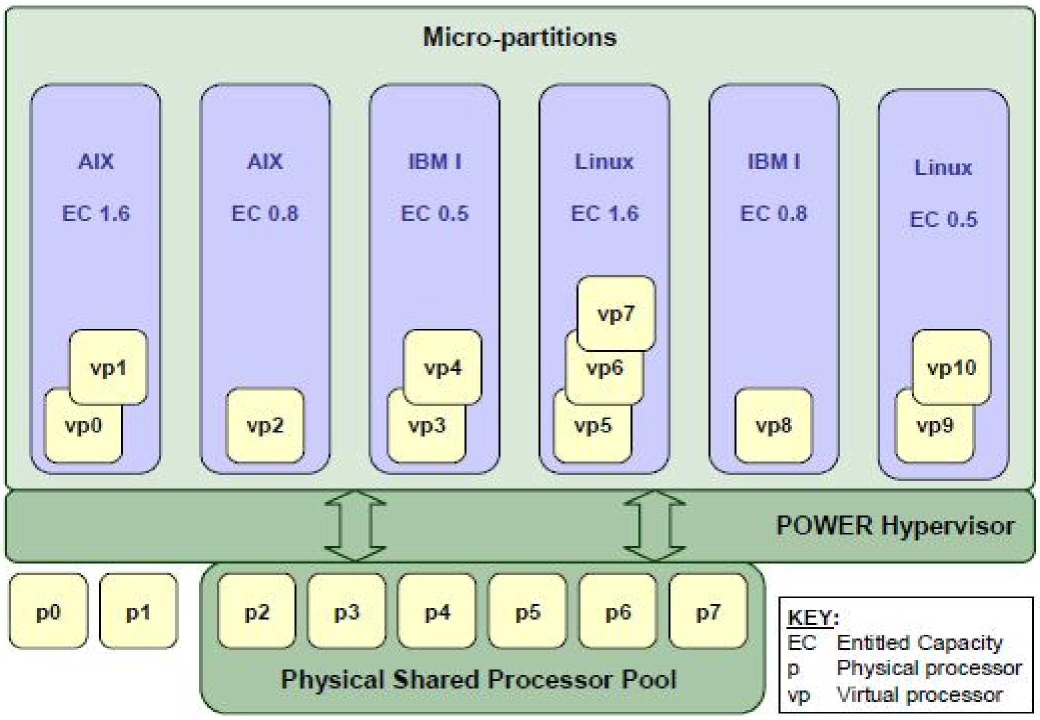


Figure 3.6 Power Physical Shared Processor Pool and micro-partition [3]

Physical processor pool consisting of processor p2, p3, p4, p5, p6, p7 and processor p0, p1 are not assigned to physical processor pool. Processor p0 and p1 can be assigned to dedicated processor partition.

3.4.2 Active Memory Sharing

Active Memory Sharing allows multiple logical partitions to share a common pool of physical memory. It is an IBM PowerVM advanced memory virtualization technology, which provides system memory virtualization capabilities to IBM Power Server.

In an IBM Power System, the physical memory can be partitioned into multiple logical partitions in either a dedicated or shared mode. Some physical memory is assigned to a logical partition and some physical memory to a pool that is shared by other logical partitions by the system administrator. A single partition can have either dedicated or shared memory.

Memory utilization on the system can be increased effectively by exploiting the active memory sharing either by:

- Decreasing the system memory requirement, or
- By allowing the creation of additional logical partitions on an existing system.

Application areas of active memory sharing (AMS) include the latest trend in technology world Cloud computing. These environments can benefit from the memory over commitment provided by AMS to stipulate more virtual servers in the cloud, increasing the global system scalability and hence resource utilization.

MODES OF MEMORY PARTITIONING

1. Dedicated Mode

The IBM Power System platform is sustaining the dedicated memory model since a long time. In this model, the physical memory is divided among the partitions with some memory dedicated to the hypervisor for maintenance of internal data structures. Any remaining memory goes to the free pool, as depicted in Figure 3.7.

During partition definition, the minimum, desired, and maximum memory for the partition are also specified. The maximum value represents the maximum amount of logical memory that can be dynamically allocated to the partition. The minimum memory value represents the minimum amount of memory required to activate the partition. The desired memory states the amount of memory that will be allocated for the partition if available. The desired memory ranges between the minimum and the maximum memory values.

System admin optimizes the available memory distribution among logical partitions. If a logical partition's performance is affected due to memory constraints and other logical partitions have vacant memory, the admin can manually issue a dynamic memory reconfiguration. So, dynamically memory can be added or removed depending on the requirement of logical partition.

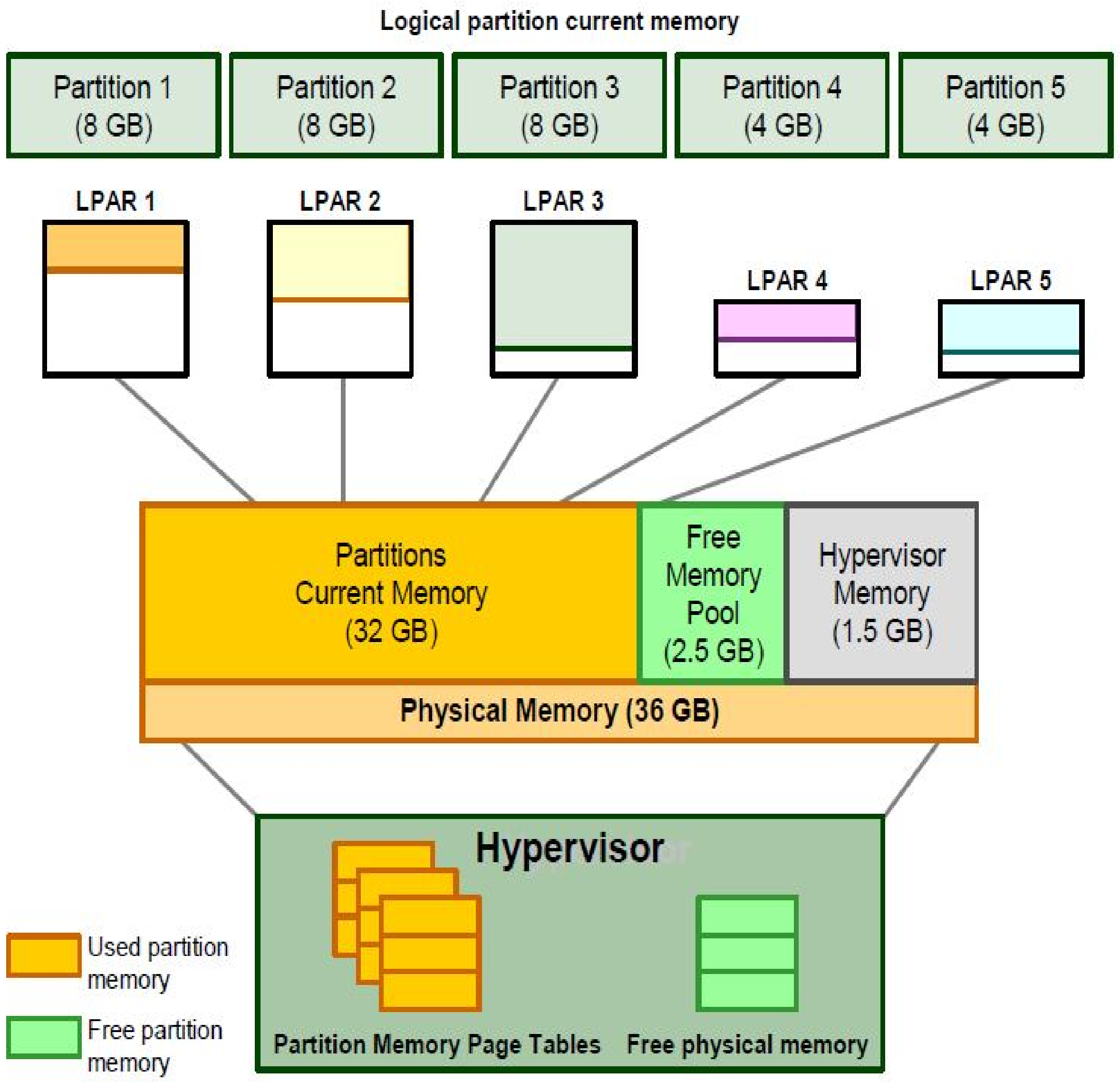


Figure 3.7 Dedicated Memory Model [2]

The configured memory is the same as the physical memory in a dedicated memory partition model. The memory partition is allocated an amount of memory greater than or equal to the minimum value specified in the partition definition and at the same time less than or equal to the desired value specified in the partition definition, depending on the amount of physical memory available in the system. Movement of memory among partitions or between partitions and the free memory can be achieved through application of Dynamic logical partition (DLPAR) operations on dedicated memory partitions.

2. Shared Mode

In the shared memory partition mode, system automatically decides the optimal allocation of the physical memory to logical partitions and adjusts the memory assignment based on demand for memory pages. Hereby, the admin just reserves physical memory for the shared memory pool and assigns logical partitions to the pool.

In this model, LPAR's share memory from a single pool and this LPAR's configured memory becomes logical memory. Here the hypervisor plays its role. Actually, the shared memory pool is part of the real physical memory. This is virtualized by the hypervisor to allocate physical memory to the shared memory partitions, as shown in Figure 3.8.

The size of the shared memory pool can be altered by:

- Adding memory from the free pool.
- Shrinking and moving (reallocation) memory from dedicated memory partitions.

Similar to the dedicated memory model, when a shared memory partition is defined, the minimum, desired, and maximum logical memory for the partition are also specified. In addition to these values, the shared memory weight can be changed by edition of the partition profile. When the hypervisor decides which partition receives the physical memory, the memory weight is one of the factors taken into consideration.

The Management Console calculates the I/O entitled memory based on the I/O configuration. The I/O entitled memory is the maximum amount of physical memory guaranteed to be available for I/O mapping.

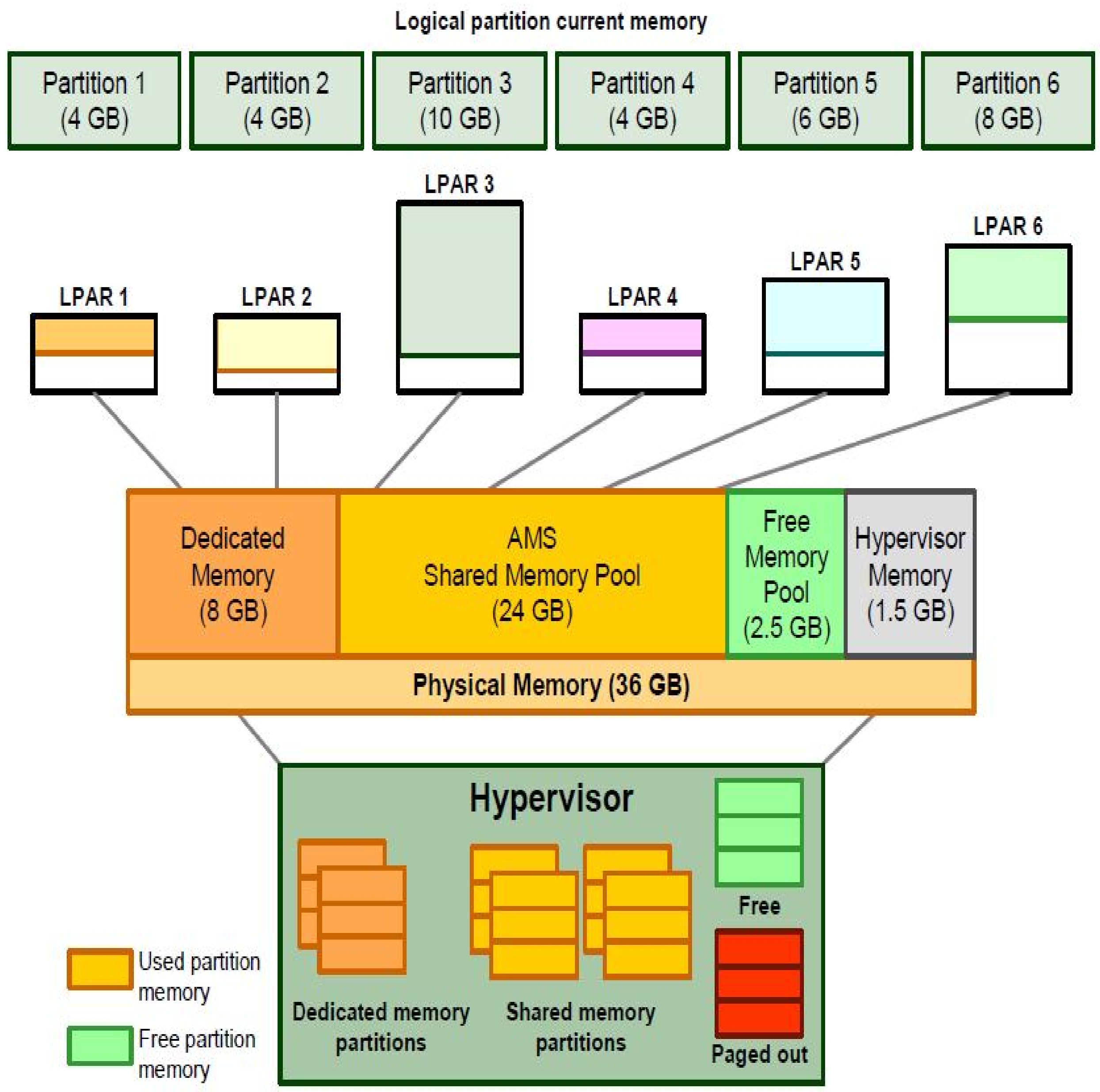


Figure 3.8 Shared Memory Model [2]

Active Memory Sharing Implementation

AMS permits selected LPARs to share memory from a single pool of physical memory. To support this function, a new level of abstraction is maintained which is managed by the hypervisor. Figure 3.9 shows the components involved in Active Memory Sharing.

An analogy exists between Active Memory Sharing support and page-based virtual memory support of traditional operating system where processes are presented with virtual address spaces that are divided into pages. These virtual memory pages can either be mapped into physical memory (page frames) or can be mapped to blocks on a disk.

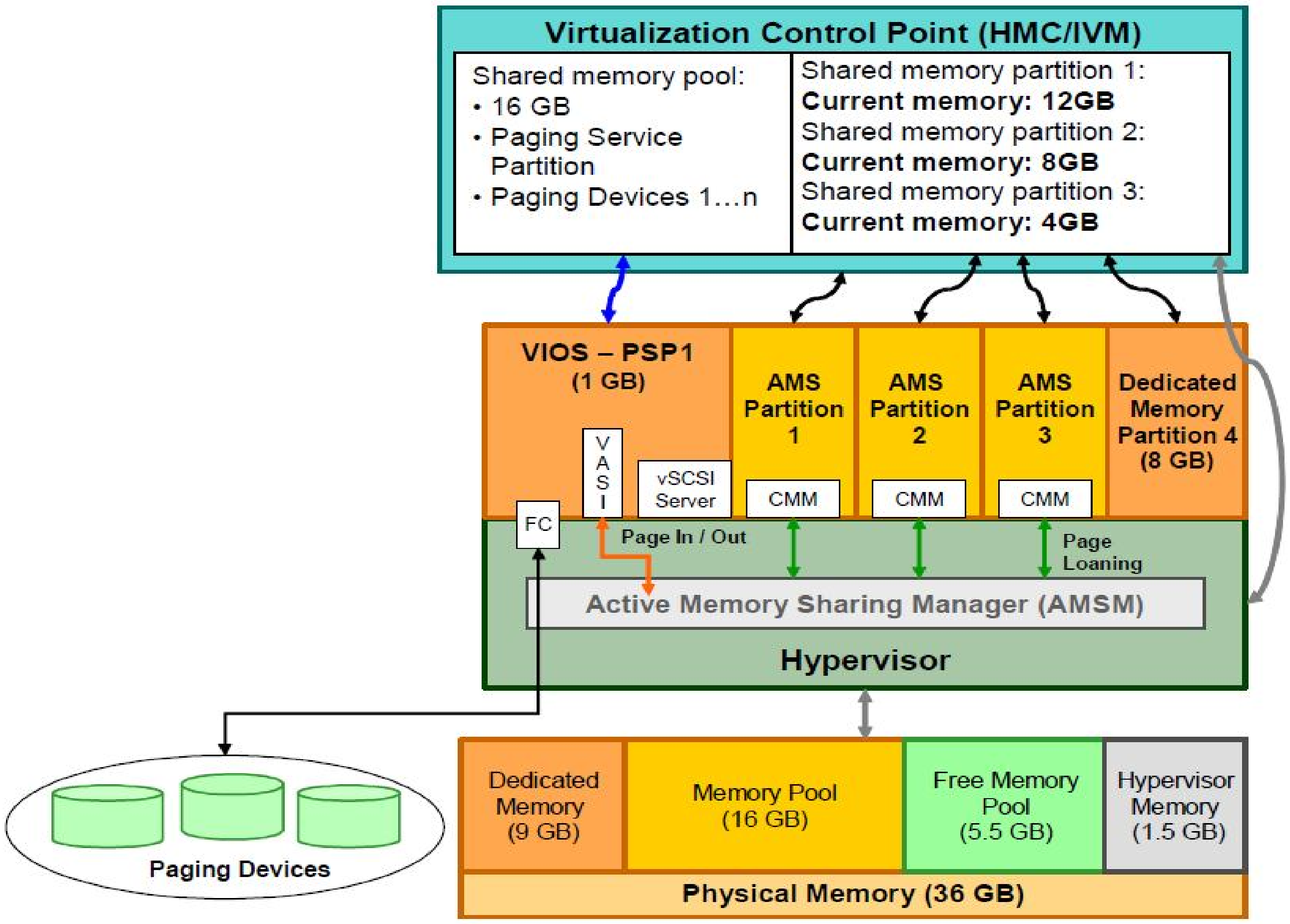


Figure 3.9 Active Memory Sharing Architecture [2]

The AMS provides the same capabilities, but in this specific case of Active Memory Sharing, the virtual address space is denoted by the LPAR's logical memory. Using Active Memory Sharing, the LPAR's logical memory is divided into virtual memory pages, and likewise the hypervisor is responsible for mapping these virtual memory pages into physical memory or to disk.

For Active Memory Sharing, the operating system assists the hyper visor for the following requests (Figure 3.10):

- Loan memory pages
- Steal aged pages
- Save pages content to a local paging device, frees and loans it to the hypervisor

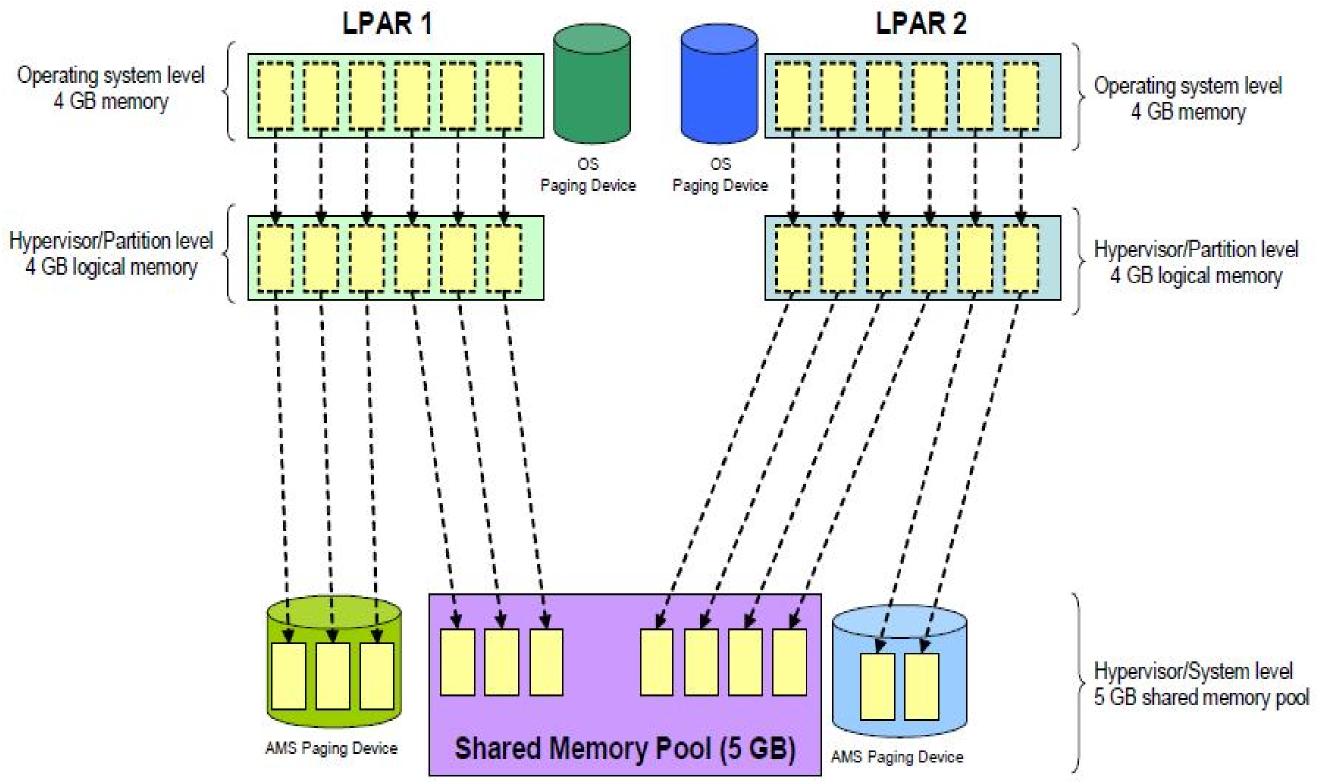


Figure 3.10: Logical Memory Mapping [2]

3.4.3 VIOS (Virtual input/output)

The VIOS forms one of the most important features of PowerVM virtualization. It is a LPAR with AIX operating system. It facilitates physical I/O resources between the LPARs.

VIOS virtualizes physical storage and network adapters. Atleast two VIOS LPARS are there in every environment. The LPAR have virtual Ethernet adapters and traffic passing through VNIC (Virtual Network Interface card) is passed to/from VIOS to/from physical adapters.

The software layer of Hypervisor (PHYP) provides a decoupling factor [6] by the virtue of which a level of indirection is introduced between the abstract (logical) and physical. This decoupling is achieved via a virtual I/O server that resides on a Linux partition. The basic idea of this decoupling is the time and space multiplexing of the available resources. There are number of benefits of using decoupling.

Some of them are-

- It provides flexible mapping between physical and logical devices with assured portability.
- It provides dynamic migration of the virtual machines from one place to another.
- Decoupling enables suspension and resumption of the virtual machine by saving its state.

One of the main goals of I/O virtualization is achieving the above mentioned benefits with minimum overhead. Number of software and hardware approaches like Para-virtualization and virtualization aware devices have been introduced to achieve high level of indirection. The scheduling and prioritization of resource becomes important during the multiplexing the requests from different VMs onto limited number of physical resources.

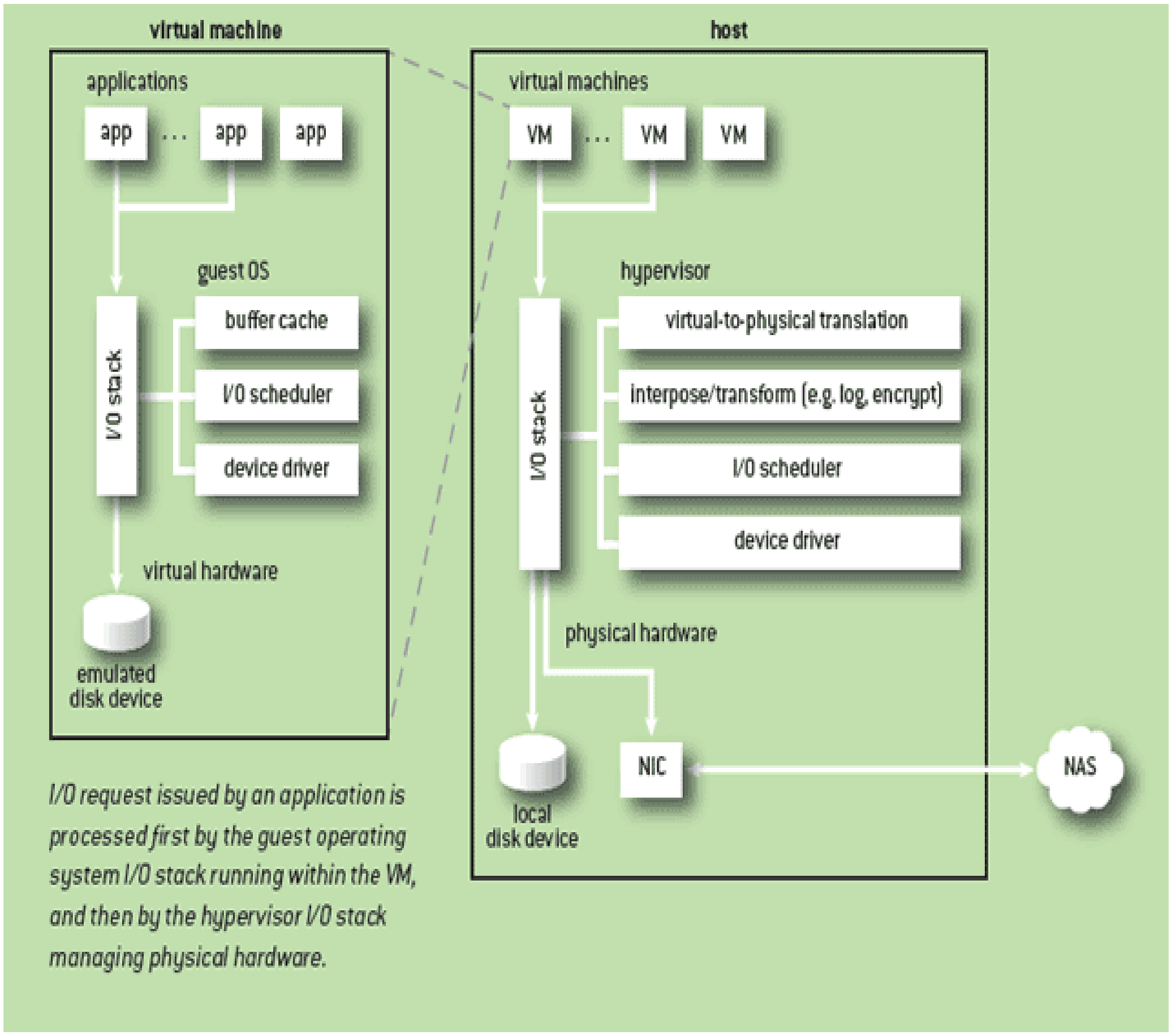

virtual machine
applications
app
app
app
guest OS
I/O stack
buffer cache
I/O scheduler
device driver
virtual hardware
emulated disk device
host
virtual machines
VM
VM
VM
hypervisor
I/O stack
virtual-to-physical translation
interpose/transform (e.g. log, encrypt)
I/O scheduler
device driver
physical hardware
local disk device
NIC
NAS
I/O request issued by an application is processed first by the guest operating system I/O stack running within the VM, and then by the hypervisor I/O stack managing physical hardware.

Figure 3.11: Processing an I/O request from Virtual Machine [6]

Processing of an I/O request raised by a virtual machine is explained in Figure 3.11. When an application running on virtual machine raises an i/o request (that is usually made by making a system call), it is initially processed by I/O stack of the guest operating system. After that via device driver present on the virtual machine, it tries to interact with a virtual device. Now this interaction request is intercepted by the Hypervisor which in turn schedules requests from multiple VMs on the physical device via another device driver managed by Hypervisor.

After that the two I/O stacks are again traversed in reverse order and finally the physical device generates a completion interrupt. This interrupt is intercepted by the Hypervisor. The Hypervisor intercepts again that to which VM the completion is associated with, and generates an virtual interrupt and issue it to virtual device driver of VM.

The modern hypervisors use a split implementation where a virtual machine can select among different virtual device interface emulation front-ends as well as multiple different back-end implementations of the device as shown in Figure 3.12.

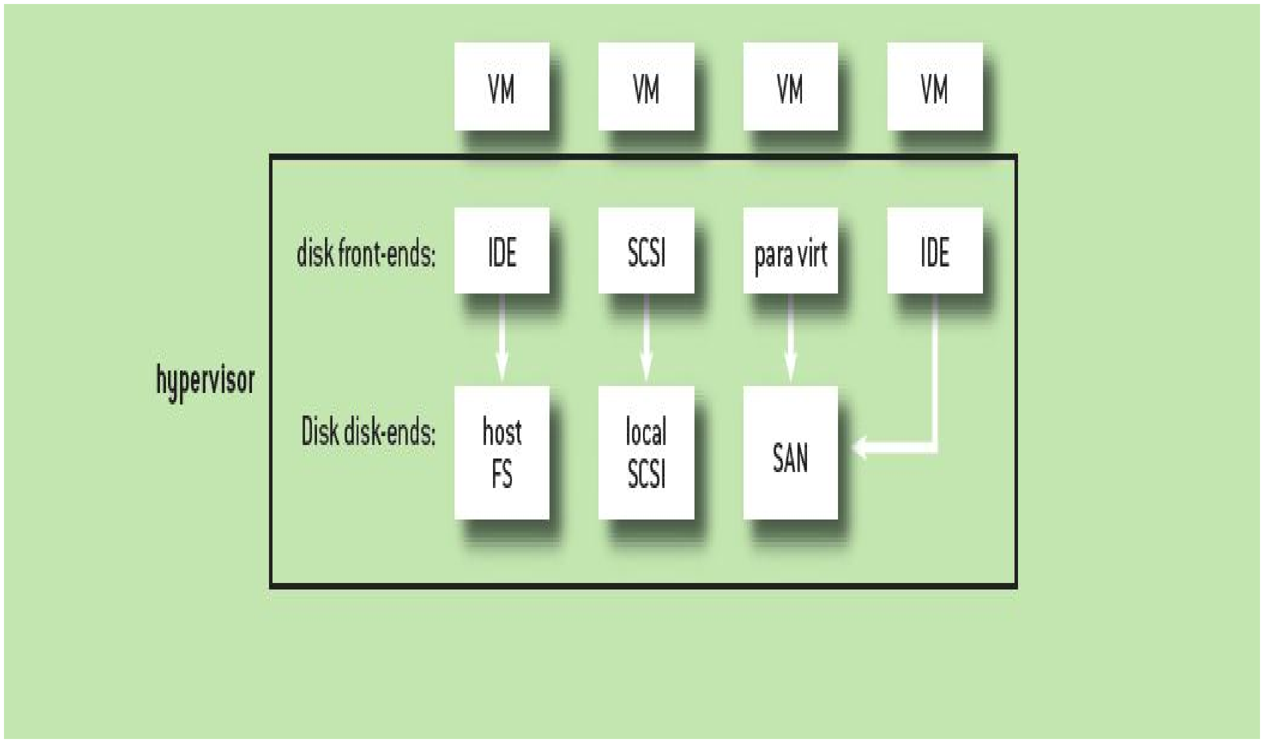


Figure 3.12: Modern Device Split Device Virtualization [6]

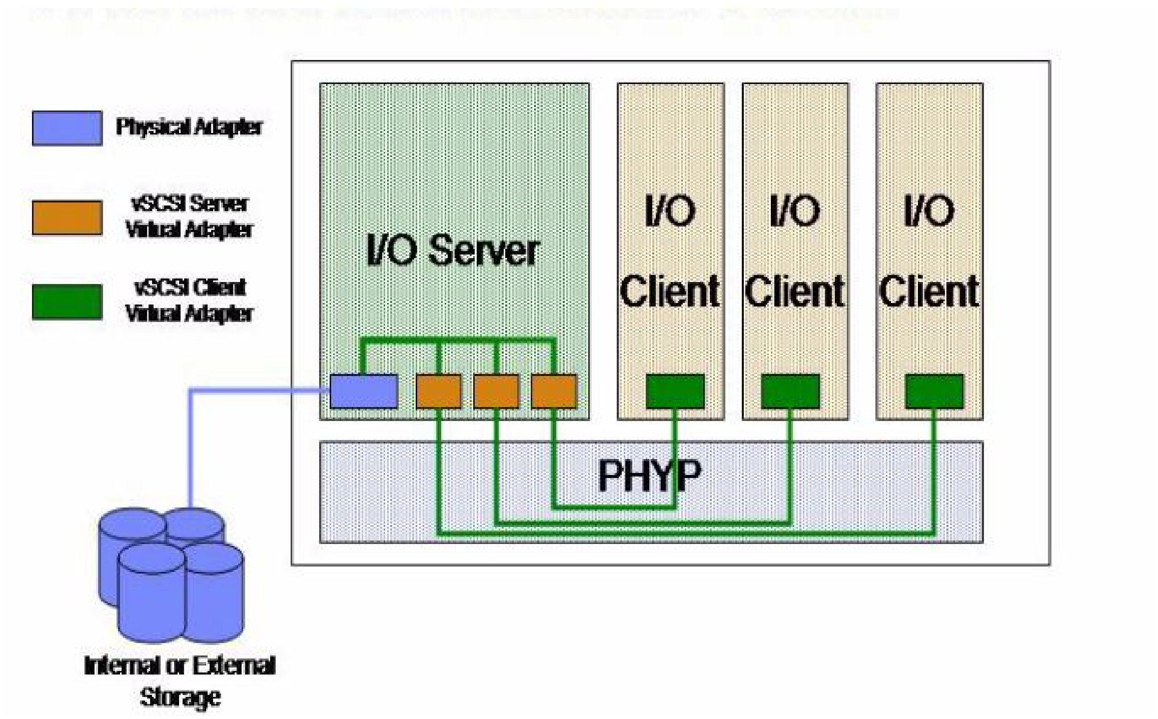
For example, the virtual machine can be configured with IDE, SCSI, or a Para-virtualized disk device that is implemented either as a file, local disk, or SAN.

While the device emulation code is specific to the particular device being emulated (e.g., an IDE disk), the semantics of the operations being performed are general and frequently constructed so that the same device emulation can access multiple different back-end implementations.

In PowerVM, the LPAR requesting a I/O operation is known as a VIO client LPAR and the LPAR containing the I/O server is known as VIOS (Virtual I/O server) (Figure 3.13). As explained already, the LPAR requesting I/O operation (VIOC) sends a request through a virtual adapter to a virtual device getting request from the virtual device driver operation.

The request for operation pertaining to the virtual device is intercepted by the hypervisor which in turn sends this request to the virtual device on the VIOS.

Now VIOS performs the multiplexing of the different requests coming from different VMs onto limited number of Physical devices via physical adapters.


Physical Adapter
vSCSI Server Virtual Adapter
vSCSI Client Virtual Adapter
I/O Server
I/O Client
I/O Client
I/O Client
PHYP
Internal or External Storage

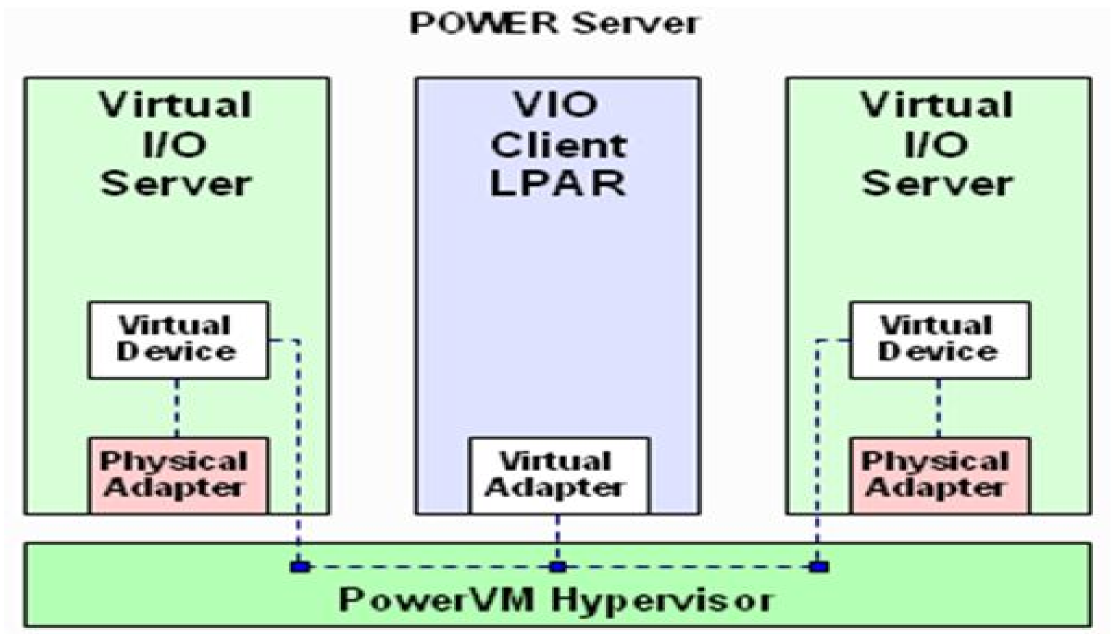


Figure 3.13: VIOS operation in PowerVM

Chapter 4 Implementation and Performance Analysis

4.1 Concept

HMC runs the management application and provides the interface for configuring and managing the virtualization features on Power Servers. As we have discussed in section 3.2, HMC communicates to PHYP via the FSP over a dedicated management network using the Secure Sockets Layer (SSL) protocol and password protection. FSP is an independent subsystem that performs system diagnostics and maintains platform configuration information. Thus eventually every command that is issued from HMC to hypervisor is routed by FSP. Hence, in Power Systems FSP is the conduit for Virtualization traffic from HMC. To improve the overall efficiency, we proposed an alternating mechanism for communication between PHYP and HMC. The mechanism reduces traffic coming towards FSP and helps in achieving following goals:

- Reduces the load on Flexible Service Processor.
- Reduces the latency of interaction with PHYP.

In the proposed architecture, we introduced a new logical partition with required capabilities of retrieving the message from HMC and redirecting it to PHYP.

4.2 Architecture

In the existing architecture all commands are routed through FSP which may or may not target PHYP. Due to this, the load on FSP is increased and the execution on PHYP is delayed. In the proposed structure, we have introduced another component i.e., a logical partition, which is used dedicatedly to route commands towards PHYP. The logical partition is created on server by PHYP and this partition hosts Linux.

Below is the proposed architecture (Figure 4.1).

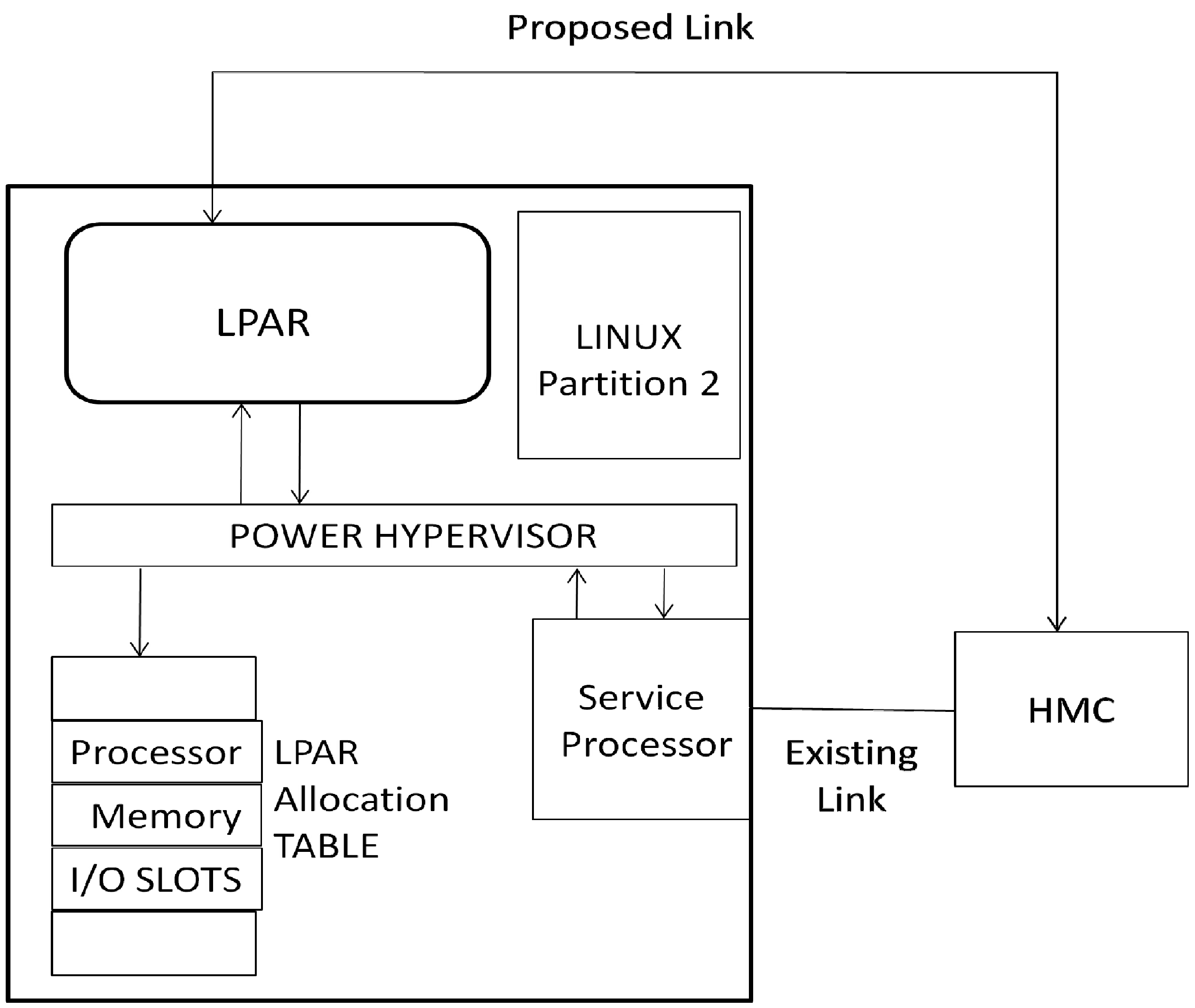


Figure 4.1: Proposed Architecture

Figure 4.1 shows how HMC interacts with FSP simultaneously with the logical partition which hosts Linux. Here the interaction between HMC & FSP aims on commands targeted for system specific tasks e.g. system diagnosis, status information etc. and command targeted to PHYP are passed on to logical partition.

A communication message structure follows standard format which is then divided into 2 parts:

- Header
- Data

The Header is of fixed size and has certain flag bits set or unset depending upon the type of message. The formative fields of the header are **opcode** and **target**. The **opcode** determines the kind of operation this message is associated with while the **target** determines the intended receiver of the message.

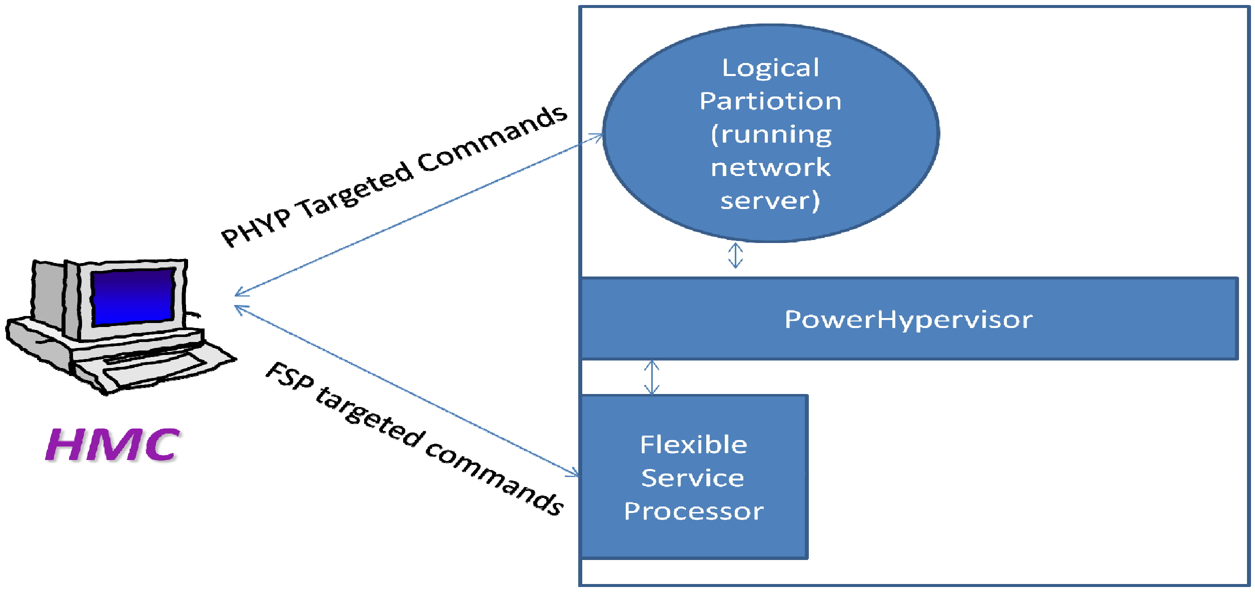

Logical Partiotion (running network server)
PowerHypervisor
Flexible Service Processor
PHYP Targeted Commands
FSP targeted commands
HMC

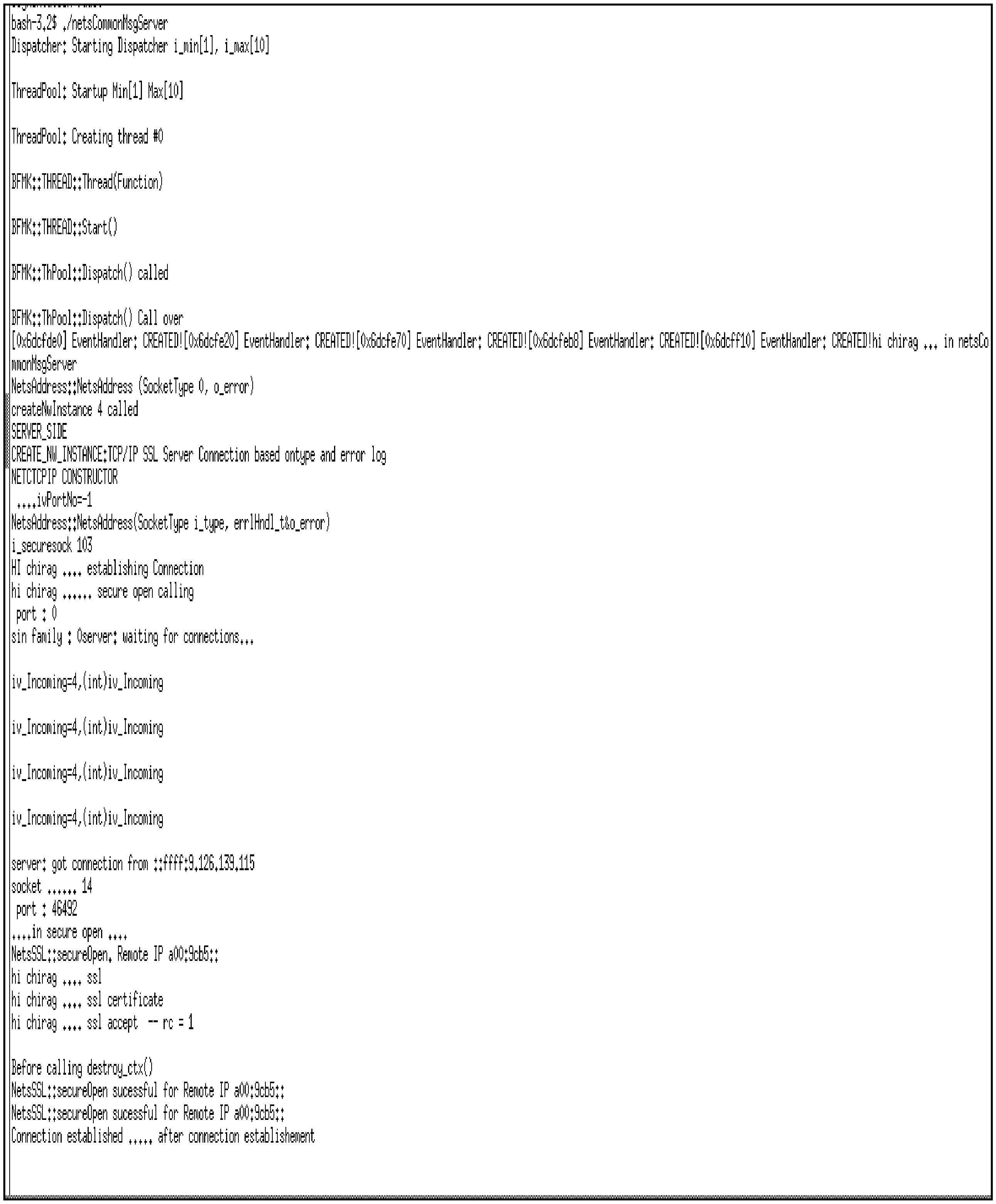
Figure 4.2 Two communication channel by HMC for interaction with Power Server

The bifurcation of the commands between the FSP and logical partition running Linux which are generated at HMC can be well depicted in Figure 4.2.

Below stated is the synopsis of implementation procedure:

- NETS (network server code) is implemented which runs on logical partition and it accepts requests coming from HMC.
- When we start NETS Server, it initializes dispatcher object providing minimum number of threads to maximum number of threads.
- NETS server calls secure open method of NETSSSL class which listens for any incoming request. When incoming request arrives at NETS server, the NETSSSL object accepts the request and sends server certificate to HMC. HMC validates the certificate and after this negotiation, NETS server creates SSL Connection.
- Following SSL connection creation, NETS server creates an object of SSLHandler and concurrently initializing it using dispatcher.
- SSLHandler object calls NETSReceive module for secure receipt of message.
- SSLHandler routes messages to ProcessMessage method of NETSProcessMsg class for processing messages.
- NETSProcessMsg class sends response back to SSLHandler.
- SSLHandler sends the response to NETSSend module. NETSSend module sends the response over a secure channel.

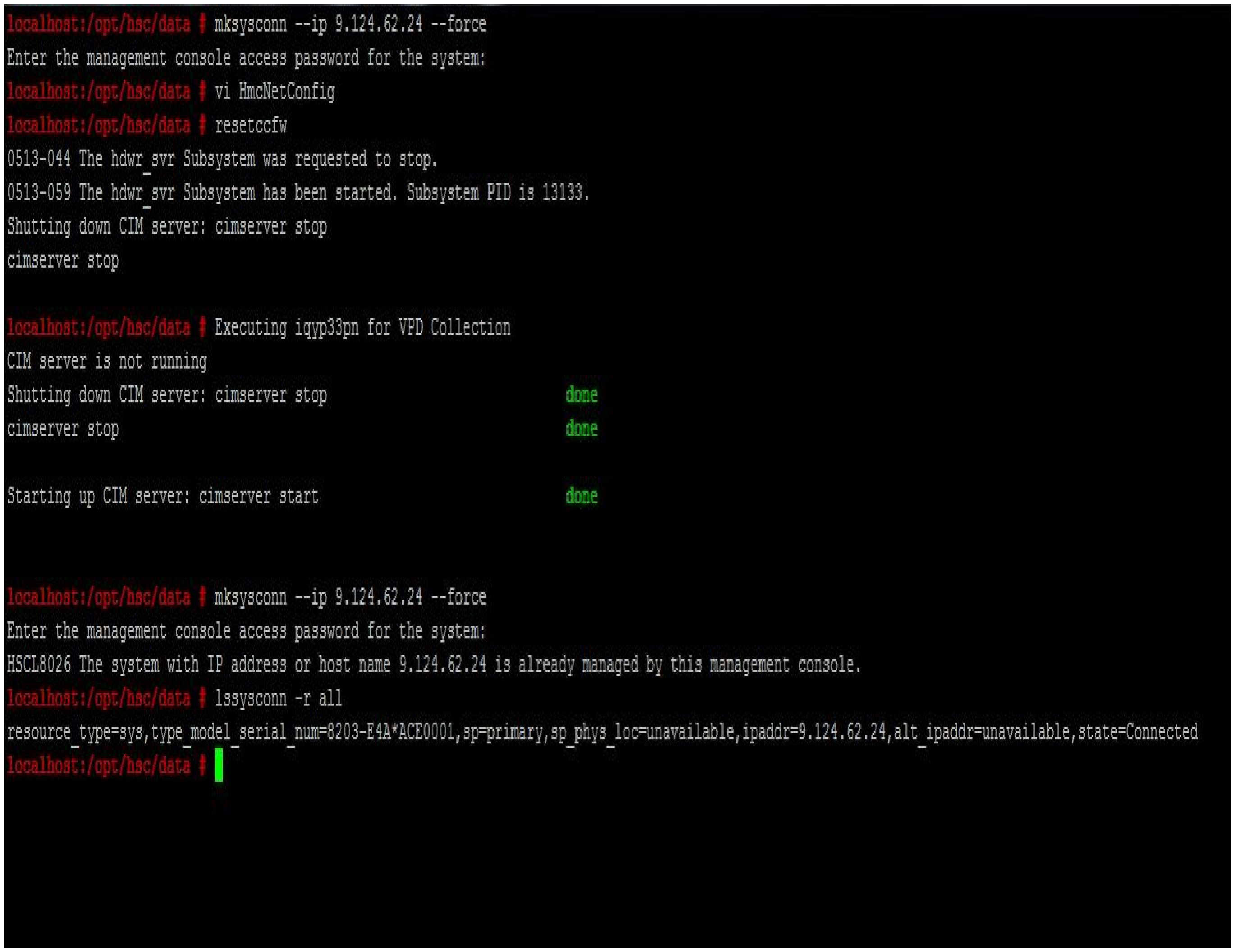
Snapshot 4.1 depicts the execution of network server on command prompt.



```
bash-3.2$ ./netsCommonMsgServer
Dispatcher: Starting Dispatcher i_min[1], i_max[10]

ThreadPool: Startup Min[1] Max[10]

ThreadPool: Creating thread #0

BFMK::THREAD::Thread(Function)

BFMK::THREAD::Start()

BFMK::ThPool::Dispatch() called

BFMK::ThPool::Dispatch() Call over
[0x6dcfde0] EventHandler: CREATED![0x6dcfe20] EventHandler: CREATED![0x6dcfe70] EventHandler: CREATED![0x6dcfeb8] EventHandler: CREATED![0x6dcff10] EventHandler: CREATED!hi chirag ... in netsCo
mmonMsgServer
NetsAddress::NetsAddress (SocketType 0, o_error)
createNwInstance 4 called
SERVER_SIDE
CREATE_NW_INSTANCE:TCP/IP SSL Server Connection based ontype and error log
NETCTCPIP CONSTRUCTOR
 ....ivPortNo=-1
NetsAddress::NetsAddress(SocketType i_type, errlHndl_t&o_error)
i_securesock 103
HI chirag .... establishing Connection
hi chirag ...... secure open calling
 port : 0
sin family : 0server: waiting for connections...

iv_Incoming=4,(int)iv_Incoming

iv_Incoming=4,(int)iv_Incoming

iv_Incoming=4,(int)iv_Incoming

iv_Incoming=4,(int)iv_Incoming

server: got connection from ::ffff:9.126.139.115
socket ...... 14
 port : 46492
....in secure open ....
NetsSSL::secureOpen. Remote IP a00:9cb5::
hi chirag .... ssl
hi chirag .... ssl certificate
hi chirag .... ssl accept  -- rc = 1

Before calling destroy_ctx()
NetsSSL::secureOpen sucessful for Remote IP a00:9cb5::
NetsSSL::secureOpen sucessful for Remote IP a00:9cb5::
Connection established ..... after connection establishement
```

Snapshot 4.1 Execution of Network Server over command prompt

Snapshot 4.2 shows the connection state between HMC and logical partition. After establishment of connection, HMC and logical partition can communicate.



```
localhost:/opt/hsc/data # mksysconn --ip 9.124.62.24 --force
Enter the management console access password for the system:
localhost:/opt/hsc/data # vi HmcNetConfig
localhost:/opt/hsc/data # resetccfw
0513-044 The hdwr_svr Subsystem was requested to stop.
0513-059 The hdwr_svr Subsystem has been started. Subsystem PID is 13133.
Shutting down CIM server: cimserver stop
cimserver stop

localhost:/opt/hsc/data # Executing iqyp33pn for VPD Collection
CIM server is not running
Shutting down CIM server: cimserver stop                        done
cimserver stop                                                  done

Starting up CIM server: cimserver start                         done


localhost:/opt/hsc/data # mksysconn --ip 9.124.62.24 --force
Enter the management console access password for the system:
HSCL8026 The system with IP address or host name 9.124.62.24 is already managed by this management console.
localhost:/opt/hsc/data # lssysconn -r all
resource_type=sys,type_model_serial_num=8203-E4A*ACE0001,sp=primary,sp_phys_loc=unavailable,ipaddr=9.124.62.24,alt_ipaddr=unavailable,state=Connected
localhost:/opt/hsc/data #
```

Snapshot 4.2: Command prompt showing HMC and logical partition connection state is connected

4.3 Impact of Architecture on Performance

There are various commands issued by HMC for different operations on servers. In existing architecture we had one communication path for interaction between HMC and power server, commands are getting directed to hypervisor from HMC via FSP. This approach overloads the processor and thereby increased the execution time.

With the implementation of new architecture having a logical partition running on server, commands issued from HMC which are targeted to hypervisor are directed via the logical partition rather FSP, we analyzed that this changed architecture results significant reduction in execution time of the commands and reduced load from FSP.

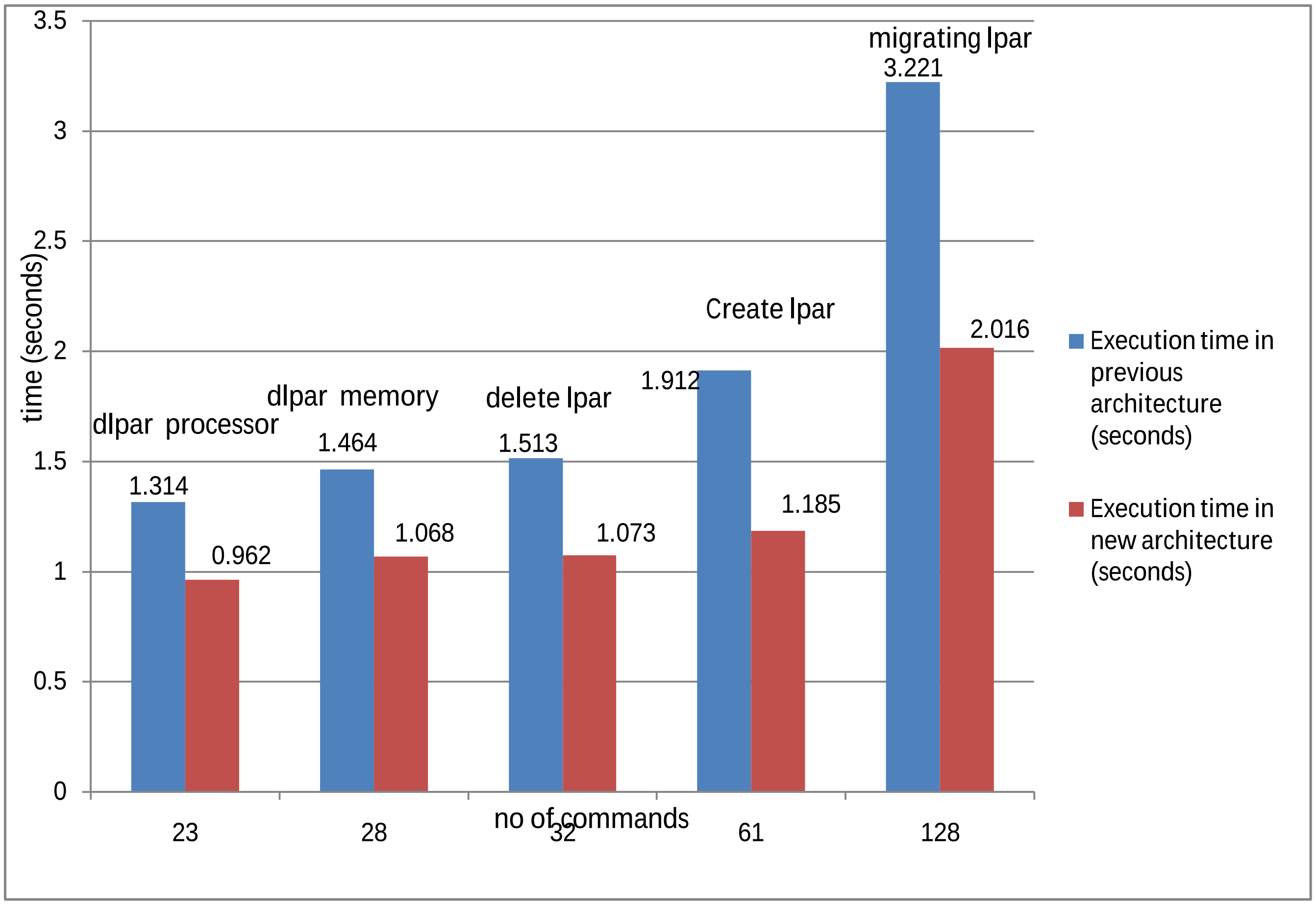


Figure 4.3: Reduction in execution times of operations

Figure 4.3 shows comparison study of the execution time of different operations (no of commands) executed from HMC to PHYP in both the architectures. Execution time depends on link capacity of both the channels. Each operation executes different number of commands for execution. We executed few operations and analyzed the no of commands for their execution with their execution times in both the architectures. As per the data analysis it is evident that the execution time for the proposed architecture has come down by significant amount.

Chapter 5 Conclusion and Future Work

5.1 Conclusion

The server virtualization technology holds a firm grip on affecting the performance of PowerVM servers. So, it's necessary to try and reduce the latency of interaction and balance the workload of components. The HMC is what a client uses to interact with the power servers and in this process of interaction, all the commands issued pass through a flexible service processor and FSP based on the command do the system configuration, check the status of server and direct the command to PHYP.

We have reduced the workload of FSP by directing PHYP targeted commands directly to a newly introduced Linux partition. On the HMC side, a decision would be taken based on the target of the message, whether to send the message to the newly introduced LPAR or to the FSP. The function of this LPAR would be to directly interact with the hypervisor and thus the workload of FSP would be reduced as well as latency of interaction would be reduced. Commands which are directed for server diagnosis, recovery and status information will be issued by HMC to FSP and commands targeted to PHYP for virtualization features will be issued by HMC to logical partition running Linux.

5.2 Future Work

We designed a special LPAR running on power servers for interaction between HMC and PHYP. There may be several cases in which running special LPAR on server may goes down. In future work, we can design an algorithm for determination of number of special LPAR can be run on power servers such that if any failure occurs with running LPAR then still we have redundant LPAR running on server such that traffic between HMC and PHYP can pass through redundant LPAR. It will improve the availability of the new communication channel between HMC and PHYP.

References

[1] Enriquillo Valdez, Renier Sailer, Ronald Perez, “Retrofitting the IBM POWER Hypervisor to Support Mandatory Access Control”, Annual Computer Security Applications Conference, pp. 221 – 231, Dec. 2007.

[2] “IBM PowerVM Virtualization Active Memory Sharing”, www.ibm.com/redbooks.

[3] “IBM PowerVM Virtualization Introduction and Configuration”, www.ibm.com/redbooks.

[4] “IBM Power Systems HMC Implementation and Usage Guide”, IBM Corporation, 2004.

[5] Maciej Mlynski, “The analysis of influence of IBM pSeries servers’ virtualization mechanism on dynamic resources allocation in AIX 5L”, International Symposium on Parallel and Distributed Computing, vol. 10, no. 2, July 2008.

[6] Mendel Rosenblum, Carl Waldspurger, “ I/O virtualization”, ACM, 2011.

[7] Jeff Daniels, “Server Virtualization Architecture and Implementation”, vol. 16, no. 1, ACM, 2009.

[8] J. Jann, L. M. Browning, R. S. Burugula, “Dynamic Reconfiguration: Basic building blocks for automatic computing on IBM pSeries Servers”, *IBM SYSTEMS JOURNAL*, vol. 42, no. 1, 2003.

[9] Narsimha Reddy CH, “Hardware based I/O Virtualization technologies for Hypervisors, Configurations and Advantages – A study”, Cloud Computing in Emerging Markets, pp. 1-5, IEEE, Oct. 2012.

[10] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, Andrew Warfield, “Xen and the Art of Virtualization”, Proceedings of the nineteenth ACM symposium on Operating systems principles, vol. 37, no. 5, pp. 164-177, ACM, Oct 2003.

[11] Rabi Prasad Pandhy, Manas Ranjan Patra, Suresh Chandra Satapathy, “Virtualization Techniques &

Technologies: State–of -the-art", vol. 2, no. 12, pp. 29-43, , Journal of Global Research in Computer Science, Dec. 2011.

[12] W. J. Armstrong, R. L. Arndt, D. C. Boutcher, R. G. Kovacs, D. Larson, K. A. Lucke, N. Nayar, R. C. Swanberg, "Advance virtualization Capabilities of POWER5 systems", *IBM J. RES. & DEV.,* vol. 49 no. 4, pp. 523-532, July 2005.