Logical Partition Security in the IBM @server pSeries 690

First Edition (02/15/2002)
Before using this information and the product it supports, read the information in "Notices" on page 9.

Contents

Introduction

The introduction of logical partitioning technology to IBM @server pSeries™ systems has greatly expanded the options for deploying applications and workloads onto server hardware. Logical partitioning (LPAR) is a server design feature that provides more end-user flexibility by making it possible to run multiple, independent operating system images concurrently on a single server. While such flexibility offers a number of desirable business advantages, it can also raise some concerns about the security implications of running operating system images in such close proximity. Of course, security has been a fundamental focus of pSeries LPAR design, as it has in the LPAR designs of other IBM @server products. The purpose of this white paper is to address these security concerns by providing an overview of the pertinent design aspects of LPAR technology on pSeries.

As this paper will illustrate, very strong isolation of operating systems running in the logical partitions comes quite naturally, due to the basic mechanisms on which logical partitioning is founded. The programs and data present in one logical partition are designed to be safe from copying or modification, whether intentional or accidental, by programs in other logical partitions. Even programming exceptions have no effect outside of the partition in which they occur. To explain this further, what follows is a brief description of the theory of operation – the "magic" – of the pSeries logical partitioning feature.

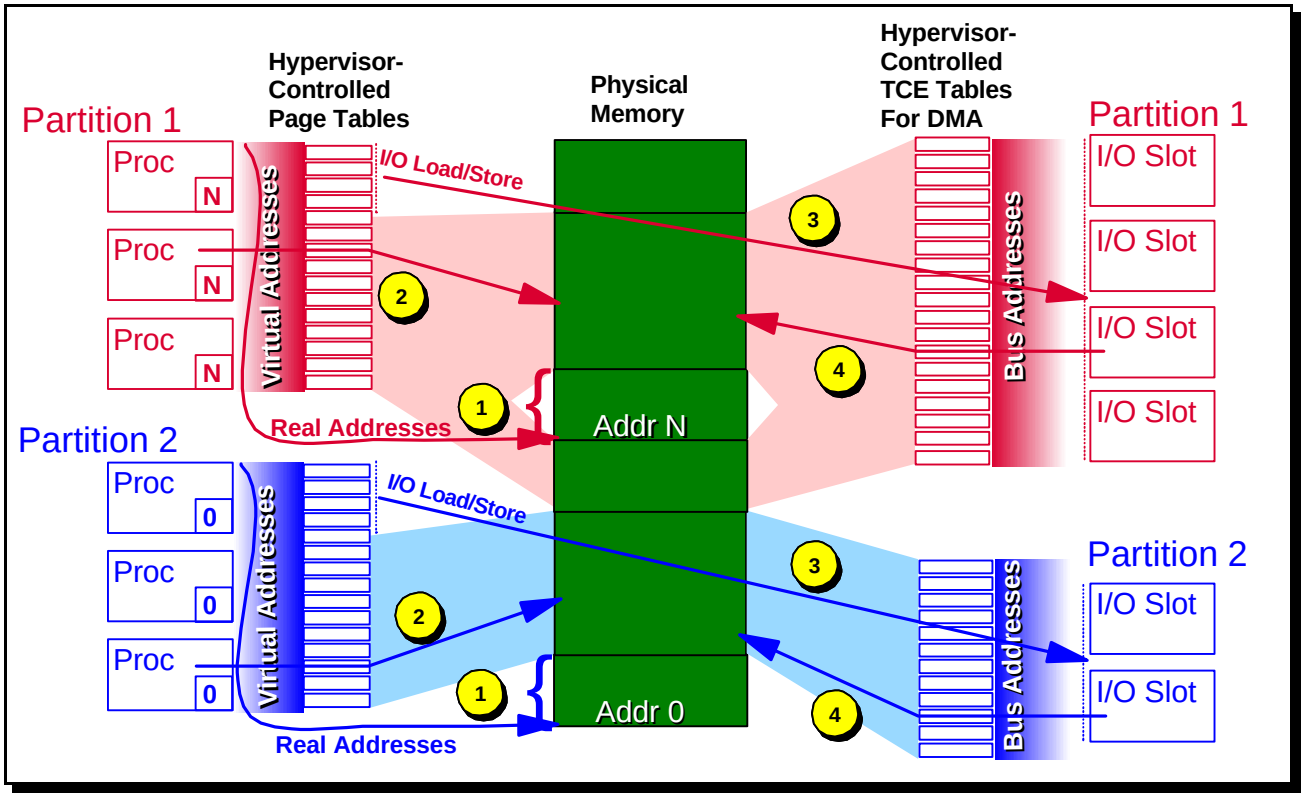
LPAR Protection in Hardware

There are actually only a few fairly simple LPAR mechanisms, implemented through special hardware registers and a trusted firmware component called a *hypervisor*. Together, these components build a tight architectural "box" around each partition, confining partition operations to an exclusive set of processor, memory, and I/O resources that have been assigned to that partition. The figure on the top of page 2 illustrates a set of address mapping mechanisms used by these resources to contain partition operations, as described in the following sections.

Real Mode Addressing

Each operating system image requires a range of memory that can be accessed in *real addressing* mode. In this mode, no virtual address translation is performed, and addresses start at address 0. Operating systems typically use this address range for startup kernel code, fixed kernel structures, and interrupt vectors. Since multiple partitions can not be allowed to share the same memory range at physical address 0, each partition must have its own real mode addressing range.

When each partition is started, firmware assigns that partition a unique real mode address *offset* and *range* value, and then sets these offset and range values into registers in each processor in the partition. These values map to a physical memory address range that has been exclusively assigned to that partition. When partition programs access instructions and data in real addressing mode, the hardware automatically adds the real mode offset value to each address before accessing physical memory. In this way, each logical partition programming model appears to have access to physical address 0, even though addresses are being transparently redirected to another address range (see arrow 1 in the figure). Hardware logic prevents modification of these registers by operating system code running in the partitions. Any attempt to access a real address outside the assigned range results in an addressing exception interrupt, which is handled by the



operating system exception handler in the partition.

Virtual Mode Addressing

Operating systems use another type of addressing, *virtual addressing*, to give user applications an effective address space that exceeds the amount of physical memory installed in the system. The operating system does this by paging infrequently used programs and data from memory out to disk, and bringing them back into physical memory on demand.

When applications access instructions and data in virtual addressing mode, they are not aware that their addresses are being translated by virtual memory management using *page translation tables*. These tables reside in system memory, and each partition has its own exclusive page table. Processors use these tables to transparently convert a program's virtual address into the physical address where that page has been *mapped* into physical memory (see arrow 2 in the figure). If the page has been moved out of physical memory onto disk, the operating system receives a *page fault*. Page translation tables are not new; they have been a part of AIX® 5L™ and pSeries server architecture for quite some time. However, page translation tables are managed differently in LPAR systems.

In a non-LPAR operation, an operating system such as AIX 5L creates and maintains page table entries directly, using real mode addressing to access the tables. In a logical partitioning operation, the page translation tables are placed in reserved physical memory regions that are only accessible to the hypervisor. In other words, a partition's page table is located outside the partition's real mode address range. The register that provides a processor the physical address of its page table is also protected by hardware logic, so that it cannot be modified by partition programs, and can only be modified by the hypervisor.

When the operating system needs to create a page translation mapping, it must execute a

specially architected *hypervisor service call* instruction on one of its processors, which transfers execution to a hypervisor program. The hypervisor program creates the page table entry on the partition's behalf and adds a logical to physical offset to the table entry before storing it. Partition programs can also make hypervisor calls to modify or delete existing page table entries. Page table entries only map into specific physical memory regions, called *logical address regions*, which have been assigned in granular chunks to that partition. These logical address regions provide the physical memory that backs up the partition's virtual page address spaces. A partition's memory, therefore, is composed of one contiguous real mode addressing region, plus some number of logical address regions, which may be assigned in any order from anywhere in memory.

Access to I/O Address Space

In addition to mapping virtual page addresses into the partition's physical memory, the operating system can make hypervisor calls to map page table addresses into the physical register and buffer address spaces on PCI I/O adapters (see arrow 3 in the figure). Device drivers read and write these adapter registers and buffers directly, which is how they set up and control I/O operations on the PCI devices. In LPAR, the PCI I/O adapter must be assigned to a given partition before that operating system can make a mapping request service call; otherwise, the hypervisor will not permit the creation of the page table entry.

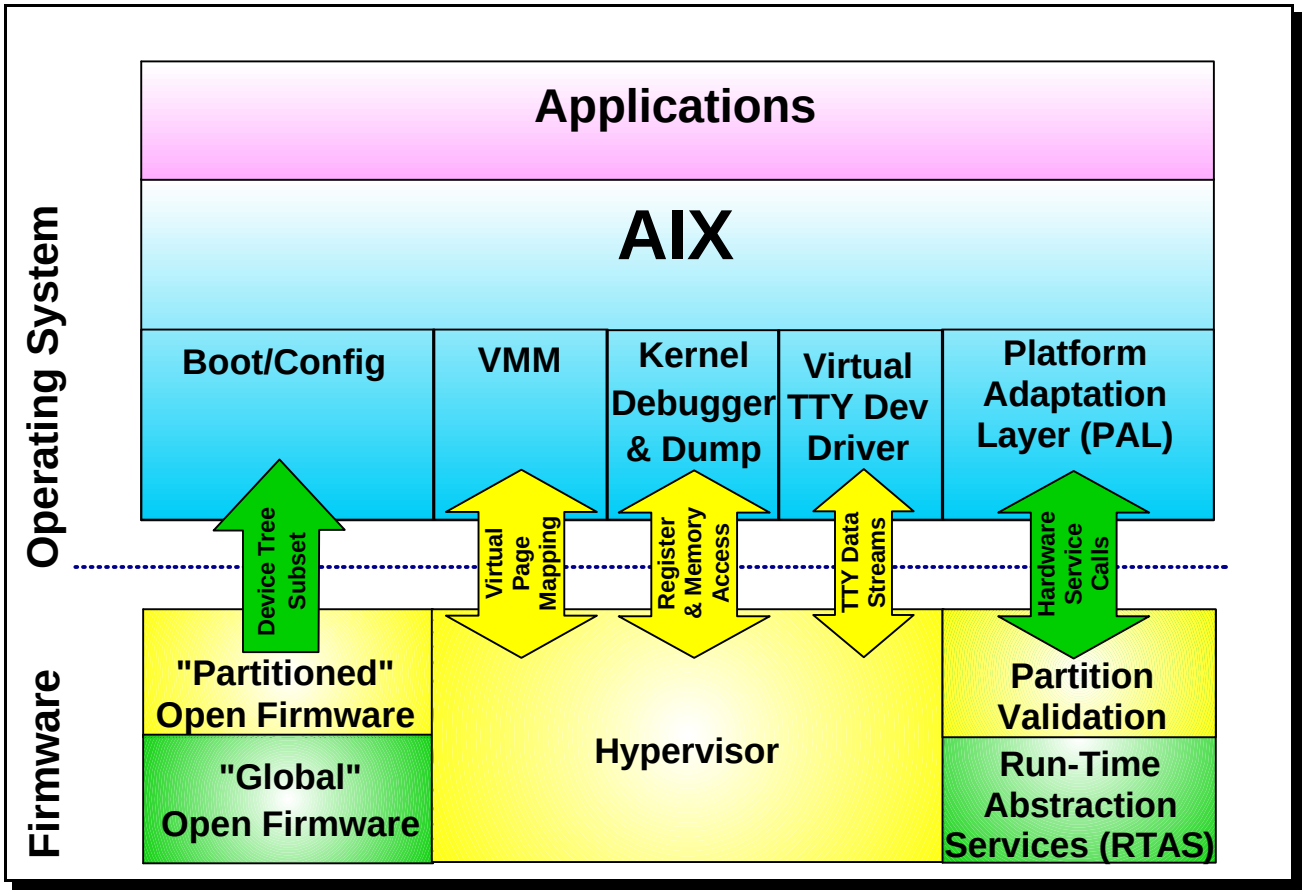
DMA Addressing

PCI I/O adapter *direct memory access* (DMA) operations move data between I/O adapters and system memory, and they use a similar address relocation mechanism to page tables. PCI host bridge hardware translates addresses generated by I/O devices into physical memory addresses. The I/O bridge makes this translation with a *translation control entry* (TCE) table, which is also stored in physical memory. As with page tables, this TCE table resides in a physical address region of system memory that is inaccessible by partitions, and only accessible by the hypervisor. Unlike page tables, however, TCE tables are not tied to any one partition. By calling a hypervisor service, partition programs can create, modify, or delete TCE table entries for the specific PCI adapters assigned to that partition. The hypervisor adds a physical address offset to the DMA target address provided by the partition program. When the I/O bridge translates an I/O adapter DMA address into physical memory, the resulting address falls within the physical memory space assigned to that partition (see arrow 4 in the figure).

LPAR Protection in Firmware

The hypervisor program is a very specialized program, entrusted to carefully perform all the operations previously described. The hypervisor program is stored in a system flash module in the server hardware. During system initialization, the hypervisor is loaded into the first physical address region of system memory. The hypervisor program is trusted to create partition environments, and is the only program that can directly access special processor registers and translation table entries. Partition programs have no way to access the hypervisor's instructions or data, other than through controlled hypervisor service calls that are part of the processor architecture. These protections allow the hypervisor to perform its duties in a simple and rigorous manner, resulting in the confinement of each operating system to a very tight, inescapable box.

In short, all access by processors and PCI I/O adapters assigned to a partition are controlled by address relocation mechanisms, and the



mechanisms themselves cannot be tampered with by partition programs, not even by firmware that runs inside the partitions.

Another way to view partition integrity and security is to compare the logical partitioning environment to a privileged or supervisor program environment that has been long understood in operating system models. Privileged supervisor programs are well protected from application or user programs, and privileged supervisor programs create environments where user applications cannot affect one another; that is, cannot view or modify another program's instructions or data. The logical partitioning hypervisor is intrinsically protected and creates multiple, isolated, high-integrity supervisor program environments.

LPAR and Operating Systems

This might seem like a very complex environment in which to run an operating system, but the logical partition environment is nearly transparent to operating systems running in the partitions. Typically, operating systems already have a kernel function that manages virtual address translations in non-LPAR environments. This one function is affected, because it must detect that it is running in a logical partition, and substitute hypervisor calls in places where it would normally directly access page tables. No other partition programs are likely to be affected by a logical partitioning operating environment.

The figure on the top of page 4 illustrates the differences that an operating system might see when running in a partitioned environment. The major difference is in the virtual memory management functions, as previously described.

At boot time, only the subset of devices that are assigned to that partition are reported to the operating system, because it does not have access to any devices outside the partition.

Some optional hypervisor services are available to obtain debug information, as well as to support a virtual TTY console device driver for each partition. The virtual TTY consoles are surfaced as windows on the hardware management console (see LPAR Management on page 5).

Lastly, pSeries hardware platforms provide a set of architected firmware Run-Time Abstraction Services (RTAS) calls. In LPAR, these calls perform additional validation checking and resource virtualization for the partitioned environment. For example, although there is only one physical non-volatile RAM chip and one physical battery-powered Time-of-Day chip, RTAS makes it appear to each partition as though it has its own non-volatile RAM area, and its own uniquely settable Time-of-Day clock. Because RTAS calls run inside a partition with the operating system, even they are not allowed to access anything outside the partition without a call to the hypervisor.

Preventing Denial of Service

As has been described, the hypervisor prevents partitions from accessing resources or data owned by other partitions. However, because LPAR runs on top of an advanced symmetrical multiprocessor architecture, some resources are also implicitly shared by the partitions. The hypervisor is designed to keep partitions from using shared resources in a way that would deny or restrict access to those resources by other partitions. A key example is the hypervisor itself, which is implemented as a library of services shared by the partitions. These services are called within the context of the operating system running in each partition. Each service call is dispatched on the specific processor from which the call was made, so hypervisor calls execute only on processors owned by the calling partition. Therefore, regardless of type and frequency of hypervisor calls made by a partitioned operating system, they can have no effect on hypervisor usage or access in other partitions.

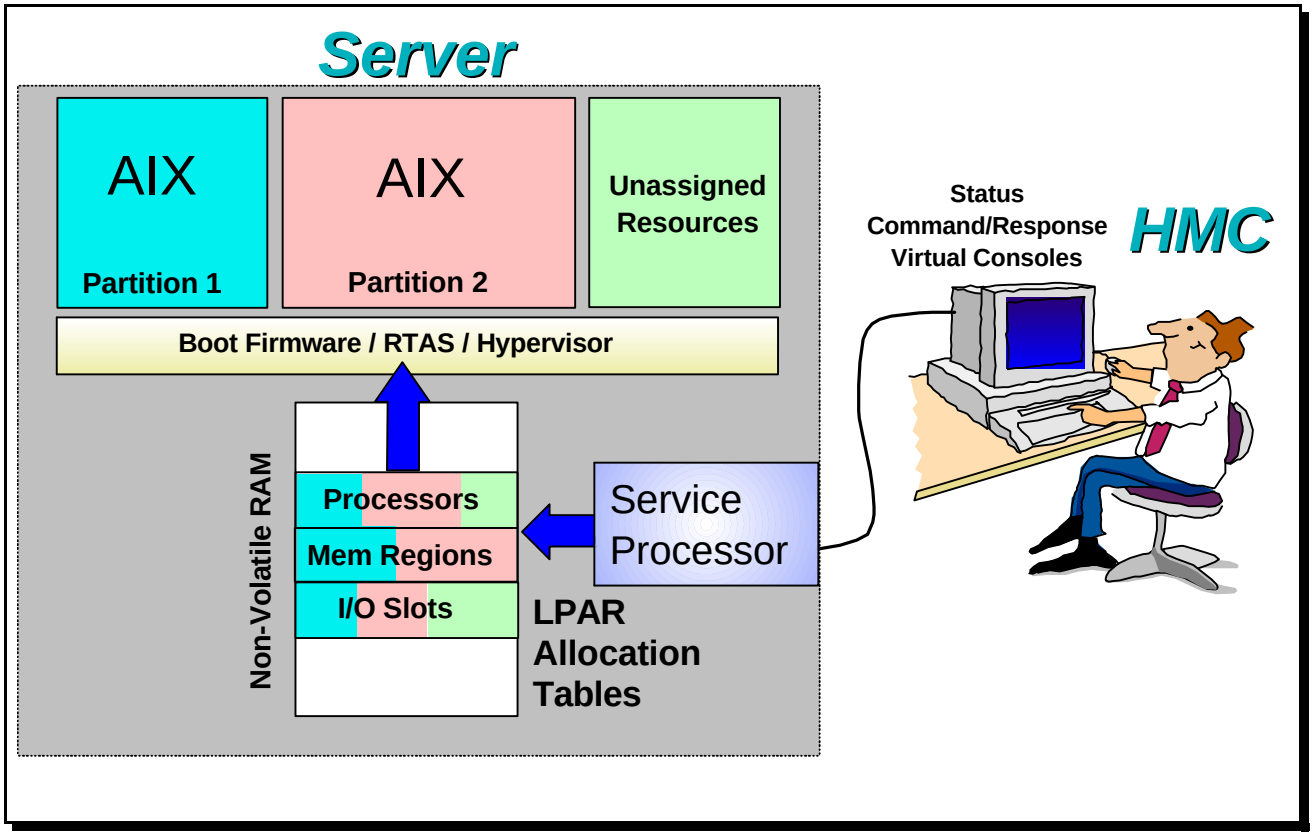
In a similar fashion, the hypervisor and hardware mechanisms protect shared hardware resources. No operation within a partition can take exclusive control of a shared hardware resource, or use a shared resource in a way that inhibits other partitions' access.

LPAR Management

The hypervisor enforces the partitioning boundaries and maintains integrity once the resources have been assigned and the partitions are up and running. To complete the security picture, it's also important to understand how partitions are managed. Along with the logical partitioning feature, pSeries has introduced the IBM Hardware Management Console for pSeries, or HMC. The HMC provides a user-friendly management appliance for performing hardware platform operations, supporting both local and remote user access. The figure on the top of page 6 shows an illustration of how an HMC works.

Connected to the service processor of the pSeries server, the HMC sends commands across this connection, which manage LPAR configuration tables that are stored in non-volatile RAM. When partitions start, the hypervisor references these tables to see which resources are assigned to which partitions. These tables are not visible to the operating systems running in the partitions, so only the HMC and the hypervisor can update them.

An HMC user must have a valid user ID and password that has been set up by the HMC



administrator. Each HMC user has an assigned role that determines the set of operations that user is allowed to do. To access the HMC functions, users can:

- **Log into the HMC locally** with their user ID and password
- **Log into the HMC remotely** over the network using a remote Web-based System Manager client, with SSL encryption if desired
- **Issue command line operations remotely** over the network, using the *rexec* command or OpenSSH

Because the HMC is shipped and packaged as a management appliance, most other network interfaces on the HMC have been disabled, to minimize any security exposures.

The HMC does also provide an application called “Service Agent” which can be used to automatically send hardware service requests to IBM through a dial-up telephone modem connection. The connection is initiated from the Service Agent application on the HMC, and it disconnects immediately after sending the service request data. Although this means that the HMC modem may be connected to a public switched network, the HMC does not permit or accept any incoming calls.

Lastly, the HMC only has access to hardware controls, and does not have any special access into the data or processes within the partitions. If desired, HMC users can be set up independently from the administrators of operating systems in the partitions. This supports a usage model where partitions are leased or given out to users other than the owner of the server hardware.

LPAR Testing and Validation

Not only does the pSeries logical partition design provide strong interpartition security and isolation based on its theory of operation, but many months of extensive lab testing have verified interpartition integrity. One test consists of running highly stressful “buster” programs in some partitions, while running surveillance programs in other partitions, and regular workloads in others. The test environment constantly monitors for any cross-partition data damage or other effects caused by the buster programs. In fact, the buster programs perform so many illegal operations that they must be constantly restarted in order to keep them running. A significant number of test hours where inter-partition integrity was maintained were accumulated prior to general availability of the logical partition feature.

The IBM @server LPAR Family

IBM @server pSeries logical partitioning is based on hardware enablement in its family of PowerPC microprocessors. This same family of microprocessors is the basis for IBM @server iSeries™, which has offered the logical partition feature since 1999.

Those familiar with IBM @server zSeries™ and S/390® logical partitioning will recognize strong similarities in the concept that all processor accesses to memory are subject to a relocation mechanism. Logical partitioning for zSeries uses an offset for every virtual address access calculation (V=F), rather than having the hypervisor create translations on a partition's behalf. In order to maximize performance, the POWER4™ microprocessor does not do this extra step on every processor translation. The tradeoff is a few more instructions executed in the hypervisor every time the operating system wants to create or modify a translation table entry, but it is still the same fundamental concept. The zSeries has been offering a logical partitioning feature for many years.

The pSeries logical partition feature does not currently offer any interpartition virtual networking or hypersockets functions found today in iSeries and zSeries logical partitioning. Similar inter-partition networking may be offered on pSeries in the future. High integrity will be maintained by the fact that the trusted hypervisor program will create and manage these inter-partition connections, only allowing connections between intended target partitions.

Logical partitioning creates many virtual computers running their own copy of an operating system. These virtual computers can also connect to common networks and can share storage devices much like stand-alone computers do today. Existing security measures relating to networks and shared storage apply to partitions in exactly the same way, and when used properly, provide a high-integrity operating environment.

Summary

The pSeries 690 LPAR feature provides increased flexibility by allowing multiple, independent operating system images to run safely and securely on a single server. This paper has explored how this environment is provided through architecture, hardware, and firmware, including:

- A **trusted firmware program**, the hypervisor, which controls access to key hardware resources
- **Hardware offset and range registers** accessible only through the hypervisor, which provide each operating system image the illusion that it has real address 0

- **Page tables**, inaccessible to the operating system and updated through hypervisor calls, to constrain virtual address access within the physical memory assigned to that partition
- **TCE tables** similarly managed to provide secure adapter DMA access
- A secure **Hardware Management Console** for management of the p690 complex
- An exhaustive **validation and test suite** to verify the integrity of LPAR design and implementation

In conclusion, the IBM @server pSeries 690 LPAR feature can be used with confidence for secure and safe server consolidation, providing information technology advantages for competitive and challenging business environments.

Special Notices

guarantee the accuracy of any information presented after the date of publication.

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

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

The [e(logo) server] brand consists of the established IBM e-business logo followed by the descriptive term “server”.

AIX, AIX 5L, pSeries, iSeries, zSeries, S/390, and POWER4. A full list of U.S. trademarks owned by IBM may be found at : http://iplswww.nas.ibm.com/wpts/trademar.htm.

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