Sharing FCP adapters through virtualization

J. Srikrishnan
S. Amann
G. Banzhaf
F. W. Brice
R. Dugan
G. R. Frazier
G. P. Kuch
J. Leopold

The IBM System z9™ and its predecessors pioneered server virtualization, including the sharing of data storage subsystems among the virtual servers of a host computer using the channel-sharing capabilities of FICON® channels in Fibre Channel (FC) fabrics. Now industry-standard Small Computer System Interface (SCSI) devices in storage area networks must be shared among host computers using the Fibre Channel Protocol (FCP), and this has been problematic with virtual servers in a host computer. To apply the power of server virtualization to this environment, the IBM System z9 implements a new FC standard called N_Port Identifier Virtualization (NPIV). IBM invented NPIV and offered it as a standard to enable the sharing of host adapters in IBM servers and FC fabrics. With NPIV, a host FC adapter is shared in such a way that each virtual adapter is assigned to a virtual server and is separately identifiable within the fabric. Connectivity and access privileges within the fabric are controlled by identification of each virtual adapter and, hence, the virtual server using each virtual adapter. This paper describes the problem prior to the development of NPIV, the concept of NPIV, and the first implementation of this technique in the FCP channel of the IBM System z9.

Introduction

This paper describes the sharing of Fibre Channel (FC) adapters by means of N_Port Identifier Virtualization (NPIV), which is a new capability of the FC standard [1]. The FC standard specifies a protocol for communication among host systems (*hosts*) and various peripheral devices (*devices*) interconnected by a network. Such networks, often referred to as *storage area networks* (SANs), comprise a set of hosts, each of which contains one or more operating system (OS) images, an interconnection network (referred to in the FC standard as a *fabric*), and devices that conform to a device protocol standard such as, for example, the Fibre Channel Protocol (FCP) for Small Computer System Interface (SCSI) standard (SCSI-FCP).

Within any network, different resources must be reserved for use by particular OS images. To accomplish this, disks and other devices conforming to the SCSI-FCP standard (*SCSI devices*) have traditionally been programmed to allow access only when a request comes from a source port address (an N_Port Identifier, or *N_Port ID*) known to belong to a specific physical port (node port, or *N_Port*) on a physical host adapter or host bus adapter (HBA), which in turn is assigned to an OS image. In addition to this use for access authorization, the host N_Port ID is the source identifier (S_ID) by which a SCSI device distinguishes among different hosts for which the device may be executing SCSI commands. The host N_Port ID is also the destination identifier (D_ID) by which a SCSI device addresses command responses back to a host. Provided that the HBAs are not shared among multiple OS images within a host, this host-identification scheme is sufficient: An N_Port ID for an N_Port on an HBA identifies the OS image using the N_Port, and the OS image is the sole user of the N_Port. (An HBA can have multiple N_Ports, but for simplicity of description here, we discuss HBAs with a single N_Port.)

With the integration of the SCSI-FCP standard into enterprise-class systems, this single-user design presents a problem. These systems often host multiple OS images,

0018-8646/07/$5.00 © 2007 IBM

and each OS image using SCSI-FCP protocols requires the presence of an unshared N_Port in an HBA in order to be uniquely identified by an N_Port ID. For example, a logically partitioned host will have multiple OS images, each requiring at least one HBA port in order to preserve the SCSI-FCP addressing model described above. This requires the purchase of a potentially significant number of HBAs, regardless of whether or not each OS image supports a workload that fully utilizes the capacity of an HBA. The host system might not even provide for the installation of the number of HBAs needed to accommodate the number of OS images the system can host, which can be more than a thousand in configurations such as an IBM System z9* with z/VM*. A solution was needed to allow the sharing of physical HBAs among the OS images, providing an addressing capability to uniquely identify each of them to the SCSI devices—or *logical units,* as they are formally known.

The sections that follow describe alternatives considered, how N_Port ID virtualization solves the problem, its implementation in the FCP channel of the System z9, and use and management considerations.

Alternative solutions

Several alternative solutions were examined; however, an underlying requirement was the need to use existing FC facilities as much as possible and to avoid changing existing upper-level protocols. Another requirement was to minimize, and if possible avoid, changes to existing network facilities.

The five alternatives that were considered for projecting the identity of OS images into the fabric are discussed in the first five sections that follow.

Using FC process associators

FC provides a capability which involves entities known as process associators that can be used to uniquely identify an OS image within a fabric. A *process associator* is a seven-byte binary number that can be chosen by the host and optionally included in the FC frame header. If there is a requirement for the host to uniquely identify each of its OS images within the fabric, it chooses a unique process associator value for each OS image and includes that process associator in all frames sent on behalf of that OS image.

Process associators would have been an ideal solution to the problem because they identify the OS image within the fabric and because they had already been defined by FC. Unfortunately, since they were not required by the FC standard, almost no existing SCSI devices support them. Therefore, although this alternative represented an attractive solution, it was not chosen because it would have precluded the use of existing SCSI devices.

Identifying the OS image at the upper-level protocol layer

The problem of projecting the identity of OS images into the network was first encountered in IBM enterprise-class systems because they were the first to provide logical partitions and virtual machines in a single physical host. They were also the first to be used with a SAN to interconnect hosts with storage devices. The first SAN I/O protocol was IBM ESCON*, which allowed sharing of I/O channels and SAN links by using a separate host logical address (HLA) at the upper-level protocol layer for each logical partition [2]. In addition, the IBM z/Architecture* and its predecessors provide the ability for an OS image to access an I/O device without necessarily being able to access other devices in the SAN, based on a host configuration definition and the system firmware. In the late 1990s, the ESCON upper-level protocol was adapted to run on FC fabrics and named FICON* [3]. The FICON protocol carries an expanded HLA over the FC fabric in the upper-level payload. This continues to be widely used today.

When smaller systems began to provide logical partitions and when SCSI devices were integrated into enterprise-class systems, the problem of identifying the OS image of a logical partition to the components of the SAN reappeared, because the SCSI-FCP standard has no HLA capability similar to that of ESCON and FICON. Thus, the first alternative considered was the addition of such an HLA capability in the upper-level protocol (i.e., SCSI). This alternative would have required fundamental modifications to the SCSI-FCP protocol, which had existed for decades, and would have required modifying a large base of existing SCSI devices. Because one of the major advantages of using SCSI devices is their low cost and extensive market penetration, having to modify them defeated the purpose of using them in the first place. Therefore, this alternative was rejected.

Using hunt groups or multicasting to get multiple N_Port IDs

Another alternative considered was the use of FC hunt groups and multicasting [1]. FC hunt groups are groups of N_Ports that are controlled by a common entity, such as an OS image. During configuration, the set of N_Ports in each hunt group is established, and each hunt group is assigned an N_Port ID that is referred to as the *hunt group identifier* (HG_ID). During normal operation, frames sent by the common controlling entity may be sent from any of the N_Ports in the hunt group using an S_ID that is (instead of the N_Port ID of the sending port) equal to the HG_ID. Similarly, frames can be sent from other N_Ports to the HG_ID, and the fabric can route the frame to any of the N_Ports in the hunt group. While the primary purpose of hunt groups is to allow data traffic to

be spread among physical N_Ports, hunt groups can be used to identify multiple OS images that share a physical N_Port by configuring each OS image to be the common controlling entity for a single-member hunt group. If, for example, there are 20 OS images sharing a single physical N_Port, that N_Port can be configured to be a member of 20 different single-member hunt groups, each with a unique HG_ID. Each OS image can then be configured as the controlling entity for one of the 20 hunt groups, thereby giving each OS image a unique HG_ID that can be used when the OS image sends and receives frames.

While this method is a possible solution and does not require any changes to existing devices or the FC protocol, it does have several disadvantages. First, only 256 hunt groups can exist in the fabric, but potentially thousands of OS images must be identified. Also, the formation of hunt groups requires an alias server in the fabric to set up the hunt groups, and there are configuration complexities in establishing the hunt groups. These complexities include the execution of a request-and-response protocol between the requesting N_Port and the alias server, as well as interaction between the alias server and the fabric in order to determine the HG_ID. Also, to prevent unwanted N_Ports from joining the hunt group, the alias server and each OS image may have to be configured with authorization passwords [4].

Another existing FC facility that was considered, multicasting, has the same disadvantages as hunt groups (i.e., being limited to 256 N_Port IDs and requiring an alias server). Also, multicasting does not allow the transfer of initiative to send frames on the FC exchange, which is a requirement of the SCSI-FCP protocol. Finally, if either hunt groups or multicasting are used, another scheme is needed to prevent additional, unwanted N_Ports from joining the multicast or hunt group. Allowing other N_Ports to join one of these groups without detection can lead to serious data security exposures. These drawbacks, which are especially important for enterprise servers, led to a search for a simpler method to solve the problem.

Emulating a subfabric with multiple N_Ports

Another category of solutions involved the virtualization of part of the fabric itself. The first FC fabrics consisted of a set of vendor-unique interconnected switches that could not communicate with switches from other vendors. As FC fabrics developed, they began to include cascaded switches (switches connected to other switches). They also began to be divided into interconnected subfabrics, so a standard was developed by which the subfabrics could communicate [5]. Given this capability, it is possible for a physical N_Port to emulate an entire subfabric containing an unlimited (except by FC addressing limitations) number of emulated N_Ports. Each OS image would be assigned an emulated N_Port ID, thereby projecting the identity of the OS image into the fabric [**Figure 1(a)**].

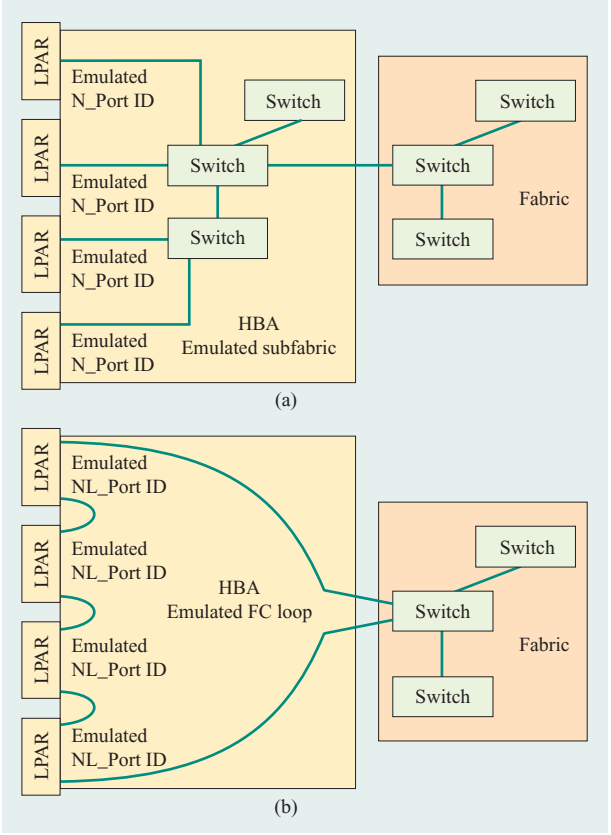

Figure 1

(a) Emulation of a subfabric with virtual switches in an HBA. (b) Emulation of an FC arbitrated loop in an HBA.

While this alternative was an attractive one on the surface, the emulation of a subfabric, especially one containing thousands of N_Ports, involves much more complexity than typical HBAs can handle. It would have required a new HBA design with the additional development time that would entail. Also, it would have required support for cascaded switches in the fabric, because the HBA would be emulating a switch.

A simplification of the fabric-emulation approach was also considered. Instead of emulating an entire subfabric, an FC-arbitrated loop (FC-AL) [6] with attached N_Ports (referred to as *NL_Ports*) and their associated NL_Port IDs [6] could have been emulated [**Figure 1(b)**].

While emulation of the FC-arbitrated loop is much simpler than emulation of a subfabric, it has a serious limitation: Only 126 NL_Port IDs can exist on a loop.

Thus, the extra development expense to modify an HBA to provide such emulation was not warranted.

Using ELS to assign N_Port IDs

The disadvantages of the ideas described above led to the investigation of the following simple yet powerful category of approaches. Although N_Ports previously had only a single N_Port ID, there is no reason why they should not be able to obtain more than one. Given this realization, a search was undertaken for the simplest way of obtaining multiple N_Port IDs while minimizing changes to existing FC protocols. As in the case of the more complex alternatives, several alternatives were considered, but they all had the common simple attribute that they queried the fabric for additional N_Port IDs.

The FC standard defines a set of services used to establish communication parameters, each of which is called an *extended link service* (ELS). An ELS comprises an ELS request sent by an N_Port and a response returned by the recipient port. One of these ELS requests, called *fabric login* (FLOGI), is sent from an N_Port to its attached fabric port (F_Port) in the adjacent switch to request the assignment of an N_Port ID. The switch responds with the N_Port ID assigned to the requesting N_Port. The first alternative investigated in this class of solutions was to use the existing FLOGI request to obtain additional N_Port IDs.

The FLOGI request is the first frame sent from an N_Port to its adjacent switch. The purpose of the FLOGI ELS is to enable the switch and the N_Port to exchange initialization parameters, including unique identifiers for the N_Port and the switch, referred to as *worldwide port names* (WWPNs), and to allow the fabric to assign an N_Port ID to the N_Port. Because the N_Port that sends the FLOGI request does not yet have an N_Port ID, it sets the S_ID in the FLOGI request to zero, and the switch responds with a FLOGI-accept response that contains the assigned N_Port ID. The HBA then uses the assigned N_Port ID as the S_ID when sending subsequent frames. It is important to note that the N_Port ID assigned to a given N_Port may change each time the N_Port is reinitialized and performs the FLOGI ELS, but the WWPN of the N_Port does not change. These characteristics allow the fabric to more effectively manage N_Port ID assignments among its ports while at the same time providing for persistent and repeatable recognition of the identity of an N_Port (i.e., its WWPN) regardless of the physical fabric port to which it is attached. Because N_Ports become associated with a specific OS image, the WWPN can be used to identify the owning OS and the access privileges it requires.

One potential method of assigning multiple N_Port IDs to an N_Port, therefore, involves issuing multiple FLOGI requests, each containing a different WWPN. The only change to existing FC protocols required by this approach would be to allow the FLOGI request to be sent more than once and with a different WWPN each time. However, it was found that this approach would have required hardware changes to many existing HBAs and switches because they had implemented portions of the link-initialization protocol (including the FLOGI ELS) in hardware.

Another alternative was to use a completely new ELS to request additional N_Port IDs. While this approach could have been used, it was abandoned in favor of an approach that reused an existing, widely implemented ELS. Reusing an existing ELS was less disruptive to existing implementations because it was already implemented and was more likely to be favorably received by the standards groups since it had already been defined.

The existing ELS, *fabric discovery* (FDISC), was originally developed for FC-AL [6] but was later allowed for use in all FC configurations. The purpose of this ELS is to verify that an existing login with the fabric is still valid. The S_ID in the FDISC request is set to the S_ID assigned during the FLOGI ELS, and the adjacent switch either rejects the FDISC request (if the S_ID or login parameters are incorrect) or accepts it (if the S_ID and login parameters are correct).

In this original use of the FDISC ELS, it was always sent with a nonzero S_ID (the presumed S_ID of the sender). Because an S_ID of zero is not explicitly prohibited by the FC standard, it was possible that additional N_Port IDs might be obtained by an extension of the FDISC ELS in a manner similar to that of obtaining the first N_Port ID with the FLOGI ELS. That is, an unlimited number of additional N_Port IDs could be obtained by using the following protocol:

- The FLOGI request is sent (with the S_ID set to zero and the WWPN set to that of the first OS image requiring an N_Port ID) to request the first N_Port ID.
- The fabric assigns the first N_Port ID to the N_Port, and that N_Port ID is used by the first OS image.
- The FDISC request is sent (with the S_ID set to zero and the WWPN set to that of the next OS image requiring an N_Port ID) to request the next N_Port ID.
- The fabric assigns the next N_Port ID to the N_Port, and that N_Port ID is used by the next OS image.

The FDISC request can be repeated until all N_Port IDs have been assigned to the N_Port, or it can be executed in parallel in order to decrease the time taken to obtain all desired N_Port IDs. This protocol for obtaining additional N_Port IDs has now been

incorporated into the FC standard [1] and is referred to as *NPIV*.

An N_Port ID must be released when an OS image no longer requires it. As with the FDISC ELS, an existing ELS that had previously been used only to terminate communication between two N_Ports is now also used to manage the removal of assigned N_Port IDs from the fabric, as follows:

- A logout (LOGO) request is sent to the fabric whenever an N_Port ID is to be removed.
- The N_Port ID to be removed is specified as the S_ID.
- The fabric removes the N_Port ID and makes it available for reuse.

NPIV was chosen as the preferred solution because it makes use of the existing FDISC ELS and LOGO ELS, minimizing changes both to the existing FC standard and to existing implementations. NPIV also allows the acquisition of an unlimited number of N_Port IDs and does not suffer from the data-security and other potential drawbacks of some of the other alternatives discussed above. Because of its network transparency, NPIV can be introduced into an FC network without affecting existing I/O devices. NPIV requires support only in the host and in the adjacent (or entry) switch to which the host is connected.

Figures 2(a) and **2(b)** provide a summary of the concepts, illustrating the differences between the two SCSI-FCP models and the potential hardware reductions to be derived from introducing NPIV.

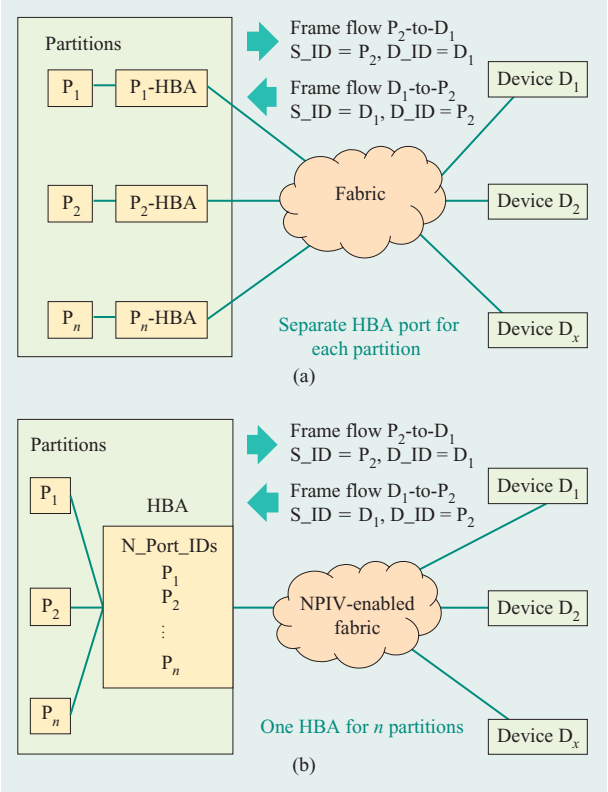

Figure 2

Host and fabric (a) without NPIV and (b) with NPIV.

NPIV as the solution in FC fabrics

Since the constructs forming the architecture for NPIV use existing extended link services, NPIV was originally included in the FC Framing and Signaling standard. Later, this standard was changed to form two FC standards: Framing and Signaling [7] and Link Services (FC-LS) [8]. Thus, it was natural for NPIV to be moved into FC-LS. Also, the FC security protocols draft standard [9] specifies that the authentication and encryption rules defined for N_Port IDs that are obtained by a FLOGI command also apply to N_Port IDs that are obtained by an FDISC command. Additional details related to the initialization of N_Ports supporting NPIV are provided in [10].

Because other N_Ports and any available fabric servers [4] are unaware that an HBA has multiple N_Port IDs, the various facilities provided by the N_Ports and servers (such as the fabric name server) are available transparently, regardless of the sharing of an HBA. For example, access to a logical unit (identified by a logical unit number, or LUN) can be controlled by what is known as a *LUN mask* in the storage subsystem such that the LUN is available for use only by permitted OS images. Similarly, the fabric name server can perform its function of providing the N_Port ID that is assigned to the WWPN of an OS image and to the other attributes of the WWPN within the fabric. Fabric zoning can be used to isolate parts of a fabric for exclusive use by a set of OS images, and state-change notifications can be used to notify other N_Ports of changes to the states of individual OS images. **Figure 3** shows an example of how zoning can be used in conjunction with NPIV to isolate subsets of the fabric for exclusive use by a given OS image. The zones circled in red and green represent subzones that the fabric can support only by using NPIV. Various enhancements to FC standards, such as FC-MI [11] and FC-GS-4 [4], have been made to support these features.

Although it is desirable to make the use of NPIV transparent throughout the fabric so that all N_Port IDs receive its benefits, SAN managers must be able to determine the physical N_Port associated with particular N_Port IDs. Various enhancements have been made to FC management protocols to make this possible. These enhancements allow SAN managers to determine the set

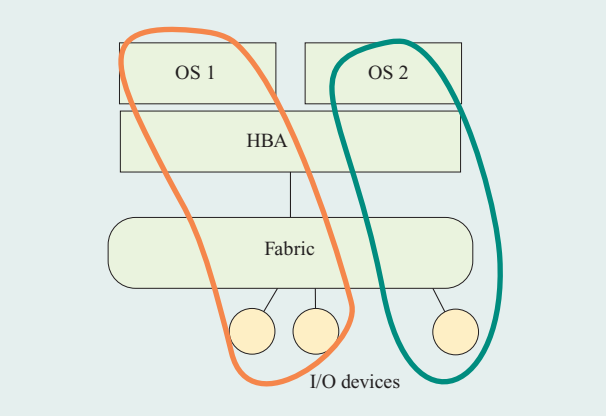

Figure 3

Fabric zoning.

of N_Port IDs assigned to an N_Port and to determine the N_Port with which any N_Port ID is associated. Additional aspects of NPIV related to SAN management are described in the use and management section of this paper.

Implementation in the System z9 FCP channel

Implementation overview

NPIV in System z9 includes allowing the assignment of multiple WWPNs to a single FC port, handling multiple fabric logins, and acquiring a unique N_Port ID to be used for subsequent I/O operations on behalf of each WWPN. The WWPNs must be predictably assigned to logical partitions and must be stored such that the assignments persist when a system is powered on and off. All of this must be done while still meeting the underlying requirements for System z9 security and its reliability, availability, and serviceability (RAS).

This section briefly describes the basic I/O concepts of the System z9 in general, the FCP channel in particular, and how the NPIV configuration and addressing methodology has been adapted to the z/Architecture. Subsequent sections describe how the additional WWPNs assigned to an FCP channel are structured and created and how the I/O subsystem is initialized for NPIV operation. Finally, the layout and handling of the central data structures used for maintaining WWPNs are described in some detail.

FCP channel and the I/O subsystem

FCP channel

The I/O component that accesses a SAN using the SCSI-FCP protocol is called an *FCP channel*. This channel type was introduced in the IBM eServer* zSeries* 900 in 2003. An FCP channel provides an FC-adapter function equivalent to that typically provided by Peripheral Component Interconnect (PCI)-based adapters on other platforms. An FCP channel also contains additional hardware and firmware to support protocol offload, enhanced RAS, and particularly shared access by multiple OS images running in various logical partitions, virtual machines, or both within the system [12, 13].

The FCP channel feature is available with all FICON Express adapters (FICON Express, FICON Express2, and FICON Express4). The FICON Express adapters provide either FCP-channel or FICON-channel functions according to the firmware with which they are loaded. The FICON Express adapter features a pair of cross-checked PowerPC* processors, a local memory, a high-speed direct memory access (DMA) engine to access system memory, and an FC HBA chipset. The firmware comprises five main components:

1. A real-time OS kernel provides the basic system infrastructure.
2. A channel and control unit component provides the interface to the z/Architecture I/O subsystem, which in turn provides I/O operations and various operations that control the configuration.
3. A queued direct I/O (QDIO) [12] component provides the primary transport mechanism for all FC traffic.
4. A layer unique to the SCSI protocol provides the programming interface to the host device drivers for SCSI devices.
5. A device driver interfaces directly with the FC HBA chipset to drive the FC link.

The FC HBA chipset, including an application-specific integrated circuit (ASIC) with associated firmware, provides a highly optimized vehicle for the transmission of SCSI commands.

System z9 made virtual with shared I/O

The IBM System z9, like its predecessors, inherits sophisticated virtualization capabilities [14]. The system provides a hypervisor function that enables the hosting of multiple logical partitions (LPARs) in which OS images can run independently. In addition, the system allows the IBM z/VM OS to run as a second-level hypervisor in one or more of these LPARs, creating virtual machines in which additional OS images can be run within an LPAR. The z/VM product provides more levels of virtualization when z/VM itself is run in a virtual machine. These virtualization capabilities allow concurrent execution of a large number of OS images on a single System z9.

This virtualization achieves its highest economic benefit by making optimum use of the I/O resources through sharing among the OS images in a controlled and secure way.

The I/O configuration of a System z9 is described in an I/O-configuration dataset (IOCDS) that defines the I/O hardware and how it is used. In particular, the IOCDS defines which LPARs have access to each I/O channel.

If the OS image running in an LPAR is z/VM, virtual-configuration definitions are provided in the virtual machine directory for each virtual machine. Directory entries define whether and how OS images in virtual machines can access I/O devices or HBA ports connected to a channel.

Resource identification

An I/O channel is identified by a one-byte channel-path identifier (CHPID). This allows for 256 channels in what is called a *logical channel subsystem* (LCSS). A full System z9 channel subsystem may include more than one LCSS, thereby allowing more than 256 channels in a configuration. Each LCSS is identified by a channel subsystem identifier (CSSID). Each LPAR has access to one LCSS.

LPARs, which may have access to these channels, are identified by partition numbers and partition names, but within an LCSS there is also a four-bit multiple-image-facility identifier (MIFID). Since the MIFID is unique within an LCSS but not unique across all LPARs, the pair (CSSID, MIFID) is one way to identify a particular LPAR in a System z9.

The channel subsystem mechanism used by an OS image to access an I/O channel is called a subchannel. In the case of an FCP channel, each subchannel can be regarded as the OS interface to the FCP channel. An FCP channel is shared among LPARs by defining multiple subchannels for the channel and associating one or more of the subchannels with individual LPARs. In the case of virtual machines within a z/VM LPAR, each of the OS images in the virtual machines is provided with one or more subchannels for each FCP channel to be accessed.

Up to 480 subchannels can be defined per FCP channel, theoretically allowing up to 480 OS images (in logical partitions or virtual machines) to share a single physical channel. However, the maximum number of subchannels that can be operated concurrently in NPIV mode is 255 per FCP channel in the System z9. This limitation is not a constraint today because (depending on the characteristics of the I/O traffic) the actual number of OS images sharing an FCP channel typically will be less than 255 to ensure reasonable performance by each OS image. Multiple subchannels of an FCP channel can be assigned to an OS image to allow the use of different subchannels with different types of I/O devices or different kinds of I/O traffic.

Subchannels within an LPAR are identified by a two-byte device number (DEVNUM) and a two-bit subchannel set identifier (SSID). Multiple sets of subchannels can be defined within an LPAR to provide for more than 64K subchannels.

Thus, to uniquely identify a subchannel and start an I/O operation, an OS image implicitly or explicitly specifies a quintuple of identifiers: CHPID, CSSID, MIFID, SSID, and DEVNUM. The CSSID and MIFID are implicit by virtue of the I/O-configuration definition for the LPAR in which the OS is running, and the SSID and DEVNUM are explicitly specified by the OS image. These four identifiers select a subchannel which implicitly identifies the applicable single (in the case of FCP) CHPID based on the I/O-configuration definition.

Virtual adapters

With NPIV, each subchannel is assigned a unique WWPN (described in the next subsection), and therefore a unique N_Port ID is obtained during fabric login, creating a virtual adapter. The I/O-configuration definition assigns each subchannel, and hence each virtual adapter, to an LPAR. When z/VM is running in an LPAR, each subchannel may similarly be assigned to a virtual machine. In this way, OS images running in an LPAR or in a virtual machine are granted access to virtual adapters with their associated WWPNs and N_Port IDs.

Defining WWPNs for virtual adapters

A WWPN is a 64-bit worldwide-unique number with various alternative formats [1]. For a System z9 virtual adapter, the WWPN has three components. First, there is a 24-bit IEEE standard component that identifies the manufacturer of the machine—in this case, IBM. This is followed by a 17-bit machine identifier that identifies an individual System z9 server. These first two comprise the WWPN common portion. The last part, which is unique for each virtual adapter on a given System z9 server, is 23 bits in length and is called the WWPN suffix. Concatenation of the three parts provides the full 64-bit WWPN for a virtual adapter.

The machine identifier is assigned when the System z9 server is manufactured, and it is stored on the machine support element (SE). The WWPN suffixes are created (for new FCP subchannels) or updated (if necessary, for changes to existing FCP subchannels) by the system firmware for all FCP subchannels defined in the IOCDS that is selected for use when the System z9 server is started (called *power-on reset*, or POR). Because the System z9 allows changes to be made to the active I/O-configuration definition while the system is running, such

changes can also cause the creation or updating of the set of WWPN suffixes in use.

Activating NPIV

Initialization of the I/O subsystem for NPIV operation

When a system POR is performed, the 41-bit company and machine identifiers, which are common to the WWPNs of all virtual adapters in an individual System z9 server, are transferred into a system-internal memory area called the *hardware system area* (HSA).

During the first POR of a System z9 server, WWPN suffixes are defined and assigned for all FCP subchannels via a WWPN assignment table (WAT) maintained in the HSA. This is done by system firmware, which assigns an arbitrary, unique WWPN suffix to each FCP subchannel in the current I/O-configuration definition.

When the POR procedure is complete, the WAT is saved on a disk device in the SE. On a subsequent POR, the WAT is reloaded into the HSA. The WAT is checked for consistency with the IOCDS that was selected for the new POR. This check is performed because an IOCDS might have been selected that is different from the one previously used, or the one previously used might have changed in a way that affects the WAT. In the event of such differences or changes, definitions of WWPN suffixes and their associations with FCP subchannels in the WAT are updated accordingly, and the updated WAT is again saved in the SE. If the current I/O-configuration definition is changed while the system is running, the necessary changes are made to the WAT and it is saved again. Thus, both the common portion of the WWPN and the specific WWPN suffixes in the WAT (for all defined FCP subchannels) are available in the HSA whenever the system completes a POR operation. Concatenation of these two parts for a particular virtual adapter is done by the firmware in the associated physical channel, as described below.

Activating the FCP channel

During one of the later steps of a system POR, channel firmware is loaded into each FCP channel. During initialization of the channel firmware, the FCP channel performs a fabric login for the physical port by issuing a FLOGI request and is assigned an N_Port ID. Also, NPIV capabilities of the attached F_Port in the adjacent switch are determined during the fabric login.

For this initial login to the FC fabric, the FCP channel uses the WWPN assigned to the physical port of the channel, which is part of the vital product data (VPD) kept in a nonvolatile storage area of the channel. This WWPN uses another of the alternative formats specified in [1]. This WWPN and the corresponding N_Port ID that is assigned during fabric login are not assigned to any subchannel (or hence, OS image) and so are not used for I/O operations initiated through subchannels in NPIV mode. However, this WWPN and N_Port ID can be used to perform I/O operations in a compatibility mode without using the NPIV capability, as described later in the section on migration from non-NPIV to NPIV. The channel logs out this port only when the entire FCP channel is stopped and reinitialized. During this initialization of the channel firmware, the channel fetches the common portion of the WWPN from the HSA to be used as the base for constructing WWPNs for virtual adapters, as described in the next subsection.

Activating virtual FCP adapters

An OS image activates a virtual adapter by running a special channel program through an FCP subchannel, as described in [12]. This creates and initializes a set of communication queues that are subsequently used to exchange command and status information and data between the channel and the OS.

During the activation of a virtual adapter, the channel fetches the WWPN suffix for the virtual adapter from the WAT and suffixes it to the common portion of the WWPN to derive the unique WWPN for the virtual adapter. The channel uses this WWPN to perform a fabric login using the FDISC ELS. If it succeeds, an N_Port ID assigned by the fabric is returned in the FDISC command response and used subsequently in the S_ID field of the FC frame header of all messages sent on behalf of the OS image using the virtual adapter.

Each N_Port ID is registered with the fabric name server; it represents a unique N_Port to the fabric and all remote N_Ports. As each N_Port consumes fabric resources, the total number of ports supported by a fabric is vendor-specific. While the WWPN for a virtual adapter is constant as long as the definition of the corresponding FCP subchannel exists in the IOCDS, the N_Port ID assigned to a virtual adapter may change with each fabric login, according to the FC standard. Once the channel firmware has finished the login and registration sequence for a virtual adapter, an OS image can start communicating with the fabric and attached devices.

Typically the OS image performs a login to the remote N_Port of a SAN-attached device, such as a disk or tape subsystem, and subsequently sends I/O requests to the device. Although the N_Port ID assigned to a virtual adapter is available to an OS image, it is not inserted into the S_ID of I/O-request payloads by the OS image. Rather, the FCP channel remembers the N_Port ID assigned to a subchannel and the corresponding virtual adapter. When the OS image sends a request, the FCP channel forms the I/O-request payload for the FC fabric and inserts the N_Port ID into the S_ID field. This is important from a security standpoint because it prevents

OS images from spoofing S_IDs, and thus from attaining unauthorized access to storage subsystems.

The FC HBA chipset includes specific support for NPIV. This HBA chipset provides a mechanism for pairing a local N_Port ID and a remote N_Port ID. The HBA chipset firmware is flexible in this regard; each pairing can be between any N_Port ID within the FCP channel and any accessible remote N_Port ID. The current implementation allows 512 such pairings. When an OS image triggers a login to a remote N_Port, the FCP channel sends a port login (PLOGI) ELS request to the remote N_Port. The HBA chipset then generates the requested pairing. The pairings persist until the OS image terminates the connection with the device. This happens when the OS image chooses to shut down the queues associated with a virtual adapter or decides to log out from the remote N_Port. At that time, the channel firmware sends a LOGO request to the remote N_Port, and the HBA chipset deletes the pairing.

Reactivation during recovery

In the event of critical internal errors, the FCP channel may be stopped and its firmware reloaded and restarted. This recovery implies a fabric logout, a resetting of all subchannels, and a subsequent relogin for the physical N_Port (using FLOGI). This type of recovery is not transparent to OS images, which have to restart the affected subchannels and reissue any pending I/O operations.

If there is a shortage of N_Port IDs available within the fabric, a race condition can occur among OS images when the subchannels are restarted. For example, suppose there is an OS image that had not previously been able to log in to the fabric because of the unavailability of an N_Port ID for its virtual adapter. Such an OS image might achieve success with an N_Port ID from among those freed by the subchannel resets. Other OS images that had previously been active might then find all available N_Port IDs taken, so that the FDISC logins for their virtual adapters would fail. To avoid such a disruption, the FCP channel maintains a list of subchannels that have been logged in and saves this list in nonvolatile storage across a channel reset. When subchannels are started after such a reset, the channel delays the FDISC login for those subchannels that had not been logged in prior to the reset, giving those that had been logged in a chance to complete their fabric logins first.

WWPN assignment table

Purpose of the WAT

The WAT records the virtual adapters of a System z9 by the identifier quintuples (see the section on resource identification above) for the corresponding subchannels.

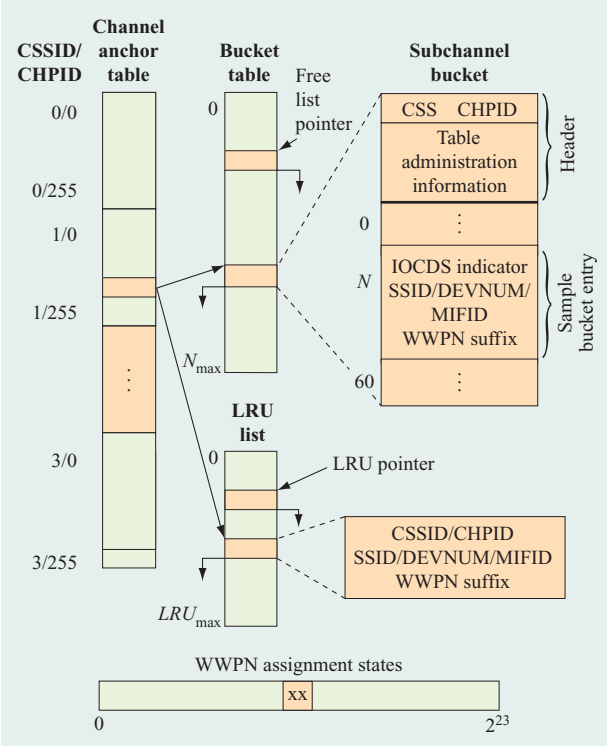

Figure 4

WWPN assignment table (WAT) structure.

A System z9 configuration—which can comprise LCSSs, channel paths, subchannel sets, device numbers, LPARs defined in an IOCDS, and WWPNs that can be assigned to a configuration—can be very complex. Therefore, the WAT is organized to avoid excessive memory consumption while still permitting access and modification with reasonable performance. This requirement results in a somewhat complex structure of interrelated tables and linked lists, which, in simplified form, is shown in **Figure 4** and explained in the following sections.

Working with multiple I/O-configuration datasets

A System z9 can have up to four IOCDSs, one of which can be selected for a given POR operation. This allows a system to operate at different times with different logical configurations for the same physical configuration, perhaps with a different subset of the I/O resources in each IOCDS or with the I/O resources assigned differently among the various LPARs.

If a System z9 is started with a different IOCDS from that of its last operational state and the new IOCDS contains some of the same FCP-subchannel definitions as the previous IOCDS (i.e., they have the same identifier quintuples), the system firmware assumes

that these subchannels serve the same purpose in both configurations and thus grants the same access rights within the SAN as it allowed their earlier counterparts. For this reason, the firmware assigns the same WWPN to such an identical subchannel in multiple IOCDSs, thereby avoiding the need to reconfigure the fabric or the storage subsystems when switching between IOCDSs.

Adding WWPN definitions

Whenever an FCP subchannel is added to the configuration to enable the creation of a virtual adapter, a WWPN must be assigned. New FCP subchannels can be added to an IOCDS before POR or, using the System z9 dynamic I/O-configuration function, they can be added to the active configuration while the system is running. In the former case, the new definition is detected during the next POR in which the IOCDS is selected and the new WWPNs are assigned during POR. In the latter case, the new WWPNs are assigned in conjunction with the dynamic configuration change. In both cases, corresponding entries are added to the WAT.

One of the types of data structures in the WAT is a bucket table comprising a set of elements called *subchannel buckets* (Figure 4). Each such bucket is associated with a single FCP channel, identified by a (CSSID, CHPID) pair, and can accommodate up to 60 subchannels, each with a bucket entry. Multiple subchannel buckets are linked to an FCP channel if more than 60 subchannels are defined for the channel.

Within a subchannel bucket, the bucket header contains the CSSID and CHPID of an FCP channel and some information for table administration purposes. A bucket entry for a subchannel includes the MIFID, SSID, and DEVNUM that identify the subchannel within the bucket, together with an indication of the IOCDSs in which the subchannel is defined and the WWPN suffix assigned to the subchannel.

If a subchannel is added and found to be new either during POR when the IOCDS is being processed or while the system is active when processing a dynamic I/O-configuration change, a new bucket entry is created. Otherwise, the existing entry is updated to indicate the inclusion of the subchannel in the current configuration.

New subchannel bucket entries are added following the last non-empty entry in the applicable subchannel bucket. Upon removal of a bucket entry (described in the next subsection), all entries below the one removed are moved upward to fill the gap that would otherwise occur.

All subchannel buckets, both empty and in use, are maintained through bucket tables. To locate the buckets (and hence the subchannels) of an FCP channel, a channel anchor table is provided with one entry per channel—i.e., one for each possible (CSSID, CHPID) pair. Figure 4 shows a channel anchor table for four LCSSs, and 256 channels per LCSS. An entry in the channel anchor table indicates the IOCDSs in which a channel is defined, and also points to a bucket table containing subchannel buckets for the channel. Additional buckets for the same channel, if any, are linked from the first bucket.

In-use subchannel buckets are linked together through multiple linked lists, one per FCP channel. Each linked list originates in the anchor table entry for its FCP channel. All empty subchannel buckets are part of a single linked list originating at a free-entry pointer representing the free-bucket pool.

Removing WWPN definitions

In the same way that FCP channels and subchannels can be added to the I/O-configuration definition, they can also be removed. An obvious mechanism for such a change is to simply remove the corresponding entries from the WAT. However, sometimes I/O resources are removed and subsequently added back to the configuration during system diagnostic and repair actions. When a subchannel definition returns in such a scenario, the assignment of a different WWPN would force the SAN or storage administrators, or both, to change the access-control mechanisms in the SAN, the storage subsystems, or both, accordingly.

To avoid this, WAT entries are not removed immediately from the WAT when a subchannel definition is removed, but rather are marked as "currently undefined." This is done by moving the subchannel definition from its bucket entry to a least-recently-used (LRU) list, which contains FCP subchannel definitions that have become undefined. The LRU list is also part of the WAT, with an entry for each subchannel similar to the subchannel bucket entries described above. An LRU list entry comprises the full quintuple defining the subchannel plus the WWPN suffix that had been assigned to the subchannel. The fixed size of the LRU list protects against excessive growth of the WAT. After a large number of such configuration changes, the oldest entries are removed from the list.

If a subchannel bucket entry being deleted is not the last (i.e., the bottom-most) entry in a bucket, all of the entries that follow it are moved upward in the subchannel bucket to avoid empty entries. If a bucket becomes empty, it is dequeued from its bucket queue and released into the free-bucket pool. If a newly defined FCP subchannel identified by the same quintuple is found to have been previously defined but has been deleted and is thus in the LRU list, the bucket entry for the subchannel is restored with the original WWPN suffix.

Just as all buckets associated with a particular FCP channel are linked from the anchor table entry for that channel, all subchannel entries in the LRU list are

included in a similar linked list from an anchor table entry. Additional pointers are used to represent the history of LRU table entries and are used to determine the entry to be discarded when that becomes necessary.

Reusing WWPNs

Because access rights such as fabric zoning and LUN masking are defined on the basis of WWPNs of N_Ports, it is critical to ensure that the same WWPN is never assigned to two different OS images. The System z9 implementation of WWPNs for NPIV in the FCP channel ensures that a particular WWPN is never assigned to more than one subchannel at a time. However, after a subchannel definition is removed and the corresponding WWPN becomes available for reassignment, it is possible or even likely that access rights for the WWPN will continue to exist in the SAN or storage subsystems for the previous use of the WWPN. Assigning the same WWPN to another subchannel that potentially belongs to a different OS image may give that OS image unintentional access to data. The NPIV implementation for the FCP channel helps to ensure that this does not happen. Before assigning a WWPN to a new subchannel, the channel firmware verifies that the WWPN suffix is not in the LRU list. However, even after the WWPN suffix has been dropped from the LRU list, the WWPN suffix is flagged as having been previously used and is thus not eligible for assignment to a subchannel.

For the unlikely case of eligible WWPNs being exhausted, which could be caused by an excessive number of configuration changes, an operator interface is provided by which WWPN suffixes that have been dropped from the LRU list can be released for reuse. The operator is informed of the possible consequences of such a release if the SAN and storage subsystems still contain access-rights definitions for the former uses of the WWPNs being released. The operator is also apprised of actions that should be taken to avoid such consequences.

This flagging of previously used WWPN suffixes is achieved by maintaining the assignment states of all possible WWPN suffixes in a list of two-bit indicators, one for each suffix. The four possible values of an assignment-state indicator describe a WWPN suffix that meets the following conditions:

- Has never been used (or has been released by operator action) and is thus eligible for assignment to newly defined FCP subchannels.
- Is assigned to an FCP subchannel defined in an IOCDS.
- Was assigned to an FCP subchannel no longer defined in an IOCDS and is now in the LRU list.
- Was assigned to an FCP subchannel no longer defined in an IOCDS and was dropped from the LRU list.

These assignment states are updated for a WWPN suffix at the following times: whenever it is assigned to an FCP subchannel; when the subchannel definition is removed and the corresponding subchannel information is transferred to the LRU list; when the bucket entry drops off the LRU list; and when the system operator explicitly releases a WWPN that has been previously assigned to a subchannel, thus making it eligible again for a new subchannel assignment.

Protection of the WAT

The information contained in the WAT is critical in various respects. For instance, invalid entries in the WAT could lead to granting unwanted access to data, and thus to a potentially serious data security exposure. Further, losing a WAT and regenerating it would require reconfiguring the SAN and storage subsystems with a newly generated set of WWPN values in order to establish the desired protection of fabric zoning and LUN masking. Consequently, two copies of the WAT are maintained, both protected by a cyclic redundancy check (CRC). Any change to a WAT is applied to both copies, and the CRC for each is regenerated.

Before the use of a WAT commences during POR, its CRC is checked. If it is found to be invalid, the CRC of the copy is checked. If the copy CRC is valid, the copy is used to override the primary WAT. If both are invalid, manual service might be able to effect recovery with a backup copy obtained from the alternate SE. In the remote event that no good copy of the WAT is available, operation of the FCP channels in NPIV mode is disabled until a new WAT can be made available through a POR with an empty WAT. At such a point, the WWPNs from the damaged WAT that had been placed in the fabric switches and storage subsystems would have to be removed and replaced with the newly assigned WWPNs from the POR with the new WAT.

NPIV-mode setup and display functions

Query WWPN information

Before virtual adapters can be used, the corresponding subchannels have to be defined in the I/O configuration and configured to operate in NPIV mode. Also, the WWPNs assigned to these virtual adapters must be retrieved and used to define the appropriate zoning in the fabric and LUN masking in storage subsystems. This section describes how this is accomplished and which operator panels are provided for that purpose. The operator panels are available on the SE and also on

a hardware management console (HMC) by using the HMC single-object operations task.

This discussion assumes that the FCP channels for which the NPIV mode is to be used are attached to FC switches capable of and configured for NPIV. It further assumes that the appropriate level of system firmware required for NPIV operation is installed and that the system POR is complete. During POR, unique WWPNs are assigned to all FCP subchannels defined in the I/O configuration, regardless of whether the subchannels are configured for NPIV.

SAN and storage administrators should define the appropriate access controls in the switches (i.e., fabric zoning) and storage subsystems (i.e., LUN masking) with the appropriate WWPNs for the FCP subchannels that will be used by the various OS images to be run in the System z9 server. This should be done before activating NPIV mode for any FCP channels. To do this, the administrators need the WWPNs of the FCP subchannels and their associations with the various OS images.

For this purpose, a "display assigned port names" panel is provided. It shows all of the FCP subchannels defined in the IOCDS, identified by the quintuple (CSSID, MIFID, CHPID, SSID, DEVNUM), in that order. The display shows, for each subchannel, the name of each LPAR to which the subchannel is configured, the assigned WWPN, the IOCDSs in which the subchannel is defined, and whether NPIV mode has been activated.

The panel has two buttons to control which subchannels are displayed. With "Show NPIV," only FCP subchannels in NPIV mode are displayed, while a default "Show all" setting causes all FCP subchannels to be displayed. The information from the displayed list can be used to configure SAN access rights as mentioned above. Pushing the "Transfer via FTP" button exports the displayed information to a File Transfer Protocol (FTP) server, such as a management server used to configure the storage subsystems or switches.

Configuring NPIV-mode operation

After configuring the fabric switches and storage subsystems with the WWPNs to be used by the virtual adapters, the virtual adapters can be enabled by activating NPIV mode for the applicable FCP subchannels. This is normally done on an LPAR basis for one or more LPARs at a time. To do this, an FCP channel image must first be configured offline to an LPAR. The operator panel can be used to activate NPIV mode for one or more FCP channel images, causing NPIV mode to be activated for all subchannels associated with the specified channel images. Then, after the channel images are configured online, the FCP subchannels are ready for use by the OS images in the applicable LPARs. When an OS image begins using an FCP subchannel, a virtual adapter is created.

After a WWPN has been assigned to an FCP subchannel and the subchannel definition is deleted, the WWPN is not eligible for reassignment to a different subchannel, as described in the section on reusing WWPNs above. Rather, after the WWPN is removed from the LRU list, the WWPN is locked. If the number of WWPNs that are available for new subchannel definitions should become exhausted (an unlikely occurrence in a normal operating environment), an additional operator panel is provided to unlock such WWPNs. The user can request that a certain number of WWPNs be unlocked, with the WWPNs to be unlocked displayed on the panel. The operator can verify that any obsolete access rights for these WWPNs have been removed from the SAN switches and storage subsystems. Alternatively, the operator can request the unlocking of all currently locked WWPNs. Since the unlocked WWPNs are not displayed, this kind of blanket unlocking holds the potential for data security exposures unless it is known that none of the WWPNs that might be unlocked are still configured in the SAN.

Migration from non-NPIV to NPIV mode

Before the introduction of NPIV, it was possible to share FCP channels among multiple OS images. However, all of the OS images had to use the single WWPN of the physical port of the FCP channel because it was not possible to obtain more than one N_Port ID for the N_Port. This meant that the OS images could not be identified individually by the components in the fabric and therefore could not benefit from various SAN facilities such as fabric zoning, LUN masking, and device reservations. Corresponding protection mechanisms had to be provided within the System z9 server in a manner somewhat different from standard industry practice [13].

To ease migration from previous systems, this capability is still supported. At installation, each FCP channel operates in non-NPIV mode by default. NPIV mode can be switched on (and off again if required) using the operator panel and the procedure described in the previous section. This older capability for sharing an N_Port ID is also required in cases in which the fabric does not support NPIV, and it can be used to share N_Port IDs in cases in which the number of N_Port IDs available from an adjacent switch is insufficient to support the number of OS images. The need for such use should decrease over time as switch and storage-subsystem vendors provide additional capabilities and capacities in their products.

Use and management of NPIV

NPIV allows the true sharing of FCP adapters among multiple OS images in a server, but this introduces the

complication of managing a potentially large number of identifiers (WWPNs) associated with the server. As the situation becomes recognized in the systems management community, management tools will be adapted and SAN management for shared adapters will be simplified.

As noted earlier, the identity of an OS image in an FC fabric is defined by the WWPNs associated with that OS image. This identity is used for multiple purposes. The ones of interest here are access control and segregation.

A SAN fabric segregates data traffic by means of zoning: Fabric switches divide the fabric into zones that are used to isolate data traffic that is to be kept absolutely separate. Conflicts may arise when data traffic from different server workloads peaks at the same time or when higher-priority short-duration traffic (e.g., transaction accesses to disk) is locked out by lower-priority longer-duration traffic (e.g., tape backup). Since there is as yet no single end-to-end prioritization mechanism in an FC fabric, segregation of traffic is the easiest solution. Zoning can also be used to segregate traffic for security reasons, preventing accidental or malicious access to storage that is reserved for private use by a restricted set of OS images. Zoning is typically implemented by configuring switch ports within the SAN fabric to accept or reject traffic from ports inside or outside the switch port zone, respectively.

A SAN fabric provides connectivity to a large number of LUNs, not all of which may be intended for access by all connected hosts. This situation can be exploited by malicious users, and it also creates the potential for data security problems due to errors in creating the configuration, if such errors allow accidental access to LUNs that should be excluded from a specific user's access list. Therefore, storage administrators typically use LUN masking—a capability in storage subsystems—to selectively mask LUNs from hosts. Such access control is done on a host port-name (WWPN) basis.

By providing each user (i.e., OS image) of a shared HBA with its own WWPN identity, NPIV enables the use of standard SAN-management controls such as fabric zoning and LUN masking for each such user. Commonly used SAN-management tools do not require any changes for NPIV because they already handle zoning controls in the fabric based on WWPNs. Similarly, storage subsystems already enable or disable access to LUNs based on WWPNs with LUN masking.

Users of NPIV may begin to see problems of scale in terms of SAN configuration and management. Historically, SAN configuration has always been a set of steps carried out manually. As servers, switches, and storage subsystems were installed and cabled together into a fabric, administrators performed the various tasks required to configure the units. With the advent of virtualization, the number of OS images (and hence

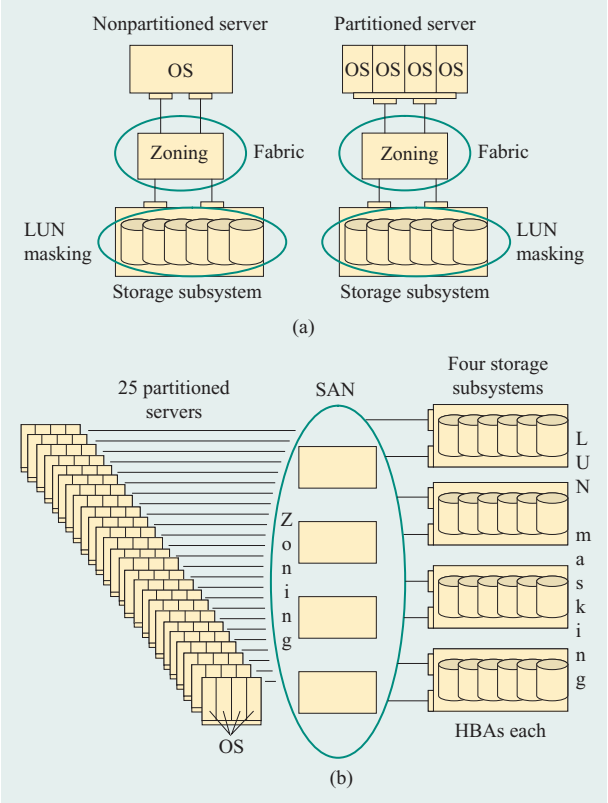

Figure 5

Zoning and LUN masking with access-controlled fabrics with two HBAs: (a) a single server; (b) 25 servers.

WWPNs) can easily become large, as does the number of LUNs required to support them, and manual methods become unwieldy.

Figure 5(a) shows two configurations. On the left is a single nonpartitioned server that must be connected to a storage subsystem. The two FC adapters (with one WWPN each) on the server must be added to the proper zone or zones of the fabric and then given access to LUNs. On the right is the same server, now partitioned with four OS images. With NPIV, there are now between four and eight WWPNs that must be added to the proper zone or zones and given access to LUNs. Four WWPNs are required if each OS image uses one of the two adapters, and eight WWPNs are required if all four OS images use both adapters.

Today's large servers are capable of supporting tens, hundreds, or even thousands of OS images in a single server. High-end storage subsystems contain thousands of LUNs. The same situation can be created with multitudes of unpartitioned, unvirtualized servers using thousands of I/O adapters, but the cost and impracticality of such a configuration has avoided these problems of scale in the

past. NPIV makes such a configuration practical and economical. When a new server is installed with NPIV, the connection of a few cables between the server and the fabric provides access to a very large number of LUNs. Each OS image has its own WWPNs. However, it can be a time-consuming and error-prone undertaking to manually configure the zoning and access controls for the shared adapters used by the large number of OS images.

Figure 5(b) shows 25 servers with four OS images each (100 OS images) that must be connected to four storage subsystems. Each server has two physical FC adapters. In this scenario, 100 to 200 WWPNs (again depending on whether each OS image has access to one or both adapters on its server) must be assigned to zones in the fabric and given access to LUNs in the storage controllers.

The manual methods do not scale up to handle the volume of configuration work required for these systems. SAN-management software can solve the problem, since it typically allows a single point of control for these tasks rather than having to establish zoning in the switches and then create access-control definitions in each storage subsystem. Today's SAN-management software was not built for such large numbers of adapters and hosts, but vendors are gradually increasing their capabilities to support IBM z/VM and other large virtualized systems.

Even with the use of improved SAN-management software, the promise of simplified provisioning of virtual servers is difficult to fulfill. In order to effectively provision a server, storage-access and fabric-zoning controls must be defined before any server can boot up, so that the server can be utilized effectively and kept within its proper boundaries as soon as it boots. Therefore, it is critical to be able to establish access control and zoning in an automated way as a part of the creation of the virtual server configuration. Currently, the server and storage-management domains are separate and distinct, making this at least a two-step task. As the problems become recognized in the industry and at standards groups, the future offers the promise of help with these management problems.

The Distributed Management Task Force (DMTF**) has created a management infrastructure based on its Common Information Model (CIM) [15]. The model defines various elements that represent a computer system and the resources it requires at both the hardware and software levels. Extensions to the model have been created by the Storage Networking Industry Association (SNIA) to enable interoperability from a management perspective among products from different vendors. The SNIA specification is called the Storage Management Initiative–Specification, or SMI-S [16]. The DMTF has recognized that virtualization requires changes to its existing models and has commissioned a working group to address these changes. Models are being defined for virtual servers and their virtual resources and for the relationships of these virtual entities to the physical entities on which they are based. These models will be integrated into the SMI-S definitions to create a seamless management interface between servers and storage.

When this work is completed and implemented in appropriate systems-management products for both servers and SANs, it will be possible in one place to define a virtual server and identify the resources it needs. Such improved systems-management products will allow the appropriate definitions to flow automatically to the right components of the SAN, allowing access to the resources needed by a given virtual server and denying access to those that are not. Provisioning a virtual server will become a simpler, one-step process.

Additionally, these models will record the relationships and dependencies between virtual and physical resources and among different types of physical resources. It will be possible to use these relationships and dependencies to do the following:

- Analyze whether a configuration has true redundancy, with no single points of failure.
- Determine the cause of congestion in a fabric and redefine the zoning to reroute traffic away from the point of congestion.
- Determine which virtual servers will be affected by preventive maintenance applied to a fabric component and whether alternate paths exist to keep the virtual server operating during the component outage.
- Increase utilization of the fabric and its components to the maximum extent feasible without losing performance due to over-utilization.
- Decrease the cost of provisioning and management by automating processes to reduce manual labor.

There are offerings on the market that do this today, such as the IBM TotalStorage* Productivity Center, which allows dynamic provisioning of both servers and storage. Such products tend to be high-end products designed for large, complex environments. With the work being done in the DMTF, such products will become more prevalent, will be designed for configurations of all sizes, and will be offered at different price points.

Summary

This paper has described the concept and implementation of NPIV as an extension to the FC standards that enables the virtualization of a physical host adapter. Each virtual adapter is assigned to a virtual server and is separately identifiable within the FC fabric. This allows the sharing

of the physical adapter while conforming to SCSI standards by allowing the control of connectivity and data-access privileges within the fabric based on the virtual server assigned to each identifiable virtual adapter. The problem of sharing FC adapters without NPIV has been described, as well as the first implementation of NPIV in the FCP channel of the IBM System z9.

The sharing of FCP adapters with NPIV is just the beginning of the simplification and resource optimization that is possible through virtualization of servers and storage. Virtualization has provided and will continue to introduce new possibilities and greater efficiencies in the use of information technology resources. Although virtualization began decades ago, today it is a technology exploding with innovation as its virtues are recognized, products embrace it, standards are adopted, and users offer wide acceptance.

Acknowledgments

The authors wish to thank the entire team that contributed to the definition of the NPIV concept and its implementation in the IBM System z9 firmware and operating systems. We also wish to thank the reviewers for their comments, which helped us improve the paper significantly.

References

1. American National Standards Institute, "Information Technology–Fibre Channel–Framing and Signaling (FC-FS)," ANSI INCITS 373-2003.
2. American National Standards Institute, "Information Technology–Single-Byte Command Code Sets CONnection (SBCON) Architecture," ANSI INCITS 296-1997 (R2002).
3. American National Standards Institute, "Information Technology–Fibre Channel–Single-Byte Command Code Sets Mapping Protocol-3 (FC-SB-3)," ANSI INCITS 374-2003.
4. American National Standards Institute, "Information Technology–Fibre Channel–Generic Services–4 (FC-GS-4)," ANSI INCITS 387-2004.
5. International Committee for Information Technology Standards, "Information Technology–Fibre Channel Switch Fabric Third Generation (FC-SW-3)," INCITS 384-2004.
6. American National Standards Institute, "Information Technology–Fibre Channel Arbitrated Loop (FC-AL-2)," ANSI INCITS 332-1999.
7. International Committee for Information Technology Standards, T11.3 Group, "Fibre Channel–Framing and Signaling–2" (draft standard under development); see *http://www.t11.org/index.htm*.
8. International Committee for Information Technology Standards, T11.3 Group, "Fibre Channel–Link Services" (draft standard under development); see *http://www.t11.org/index.htm*.
9. International Committee for Information Technology Standards, T11.3 Group, "Fibre Channel–Security Protocols" (draft standard under development); see *http://www.t11.org/index.htm*.
10. American National Standards Institute, "Information Technology–Fibre Channel–Device Attach (FC-DA), ANSI INCITS TR36-2004.
11. International Committee for Information Technology Standards, "Information Technology–Fibre Channel–Methodologies for Interconnects (FC-MI)," INCITS TR30-2002.
12. I. Adlung, G. Banzhaf, W. Eckert, G. Kuch, S. Mueller, and C. Raisch, "FCP for the IBM eServer zSeries Systems: Access to Distributed Storage," *IBM J. Res. & Dev.* **46**, No. 4/5, 487–502 (2002).
13. G. Banzhaf, R. Friedrich, S. Mueller, and C. Rund, "Host-Based Access Control for zSeries FCP Channels," *z/Journal* **3**, No. 4, 99–103 (2005).
14. L. W. Wyman, H. M. Yudenfriend, J. S. Trotter, and K. J. Oakes, "Multiple-Logical-Channel Subsystems: Increasing zSeries I/O Scalability and Connectivity," *IBM J. Res. & Dev.* **48**, No. 3/4, 489–505 (2004).
15. Distributed Management Task Force, Inc., Common Information Model (CIM) Standards; see *http://www.dmtf.org/standards/cim*.
16. Storage Networking Industry Association (SNIA), *SNIA Storage Management Initiative Specification*; see *http://www.snia.org/smi/tech_activities/smi_spec_pr/spec*.

Received March 10, 2006; accepted for publication June 16, 2006; Internet publication January 11, 2007

Jaya Srikrishnan *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (jaya@us.ibm.com).* Ms. Srikrishnan received a B.A. degree in economics from the University of Mumbai, India, and an M.S. degree in computer science from Union College. She is a Senior Technical Staff Member in the I/O-firmware development organization, working on a number of I/O-related projects on IBM System z* and System p* servers.

Stefan Amann *IBM Systems and Technology Group, IBM Deutschland Entwicklung GmbH, Schoenaicherstrasse 220, 71032 Boeblingen, Germany (stefan.amann@de.ibm.com).* Mr. Amann received a Dipl.-Ing. degree in computer science from the Berufsakademie, Stuttgart, Germany. He leads a development team working on the System z9 FCP channel. Mr. Amann is a coinventor on several patents.

Gerhard Banzhaf *IBM Systems and Technology Group, IBM Deutschland Entwicklung GmbH, Schoenaicherstrasse 220, 71032 Boeblingen, Germany (banzhaf@de.ibm.com).* Dr. Banzhaf received an M.S. degree in computer science from the Technical University of Karlsruhe and a Ph.D. degree in electrical engineering from the University of Siegen. He is working on the development of I/O subsystems and firmware for System z9, with a special focus on server and I/O virtualization. Dr. Banzhaf is a coinventor on several patents.

Frank W. Brice *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (fbrice@us.ibm.com).* Mr. Brice received a B.S. degree with honor from Stevens Institute of Technology and an M.S. degree in computer science from Rutgers University. He is a Senior Technical Staff Member in z/VM development, performing I/O-architecture and system-design work for z/VM and the System z platform. Mr. Brice has received several IBM awards and is a coinventor on several patents.

Robert Dugan *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601.* Mr. Dugan received B.S.E.E. and M.S.E.E. degrees from Auburn University. He is a retired Senior Technical Staff Member, having worked most recently as the project leader for the development of a future I/O architecture for System z9. Mr. Dugan is a coinventor on the original NPIV patent as well as several other patents.

Giles R. Frazier *IBM Systems and Technology Group, 11400 Burnet Road, Austin, Texas 78758 (grf@us.ibm.com).* Mr. Frazier received B.S.E.E. and M.S.E.E. degrees from Stanford University. He is a Senior Technical Staff Member working on the development of the new IBM PowerPC Architecture*. Mr. Frazier is a coinventor on the original NPIV patent as well as several other patents.

George P. Kuch *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (gkuch@us.ibm.com).* Mr. Kuch received a B.S.E.E. degree from Rutgers University and an M.S.E.E. degree from Syracuse University. He is the IBM Poughkeepsie team leader for System z9 FCP channel firmware development.

Juergen Leopold *IBM Systems and Technology Group, IBM Deutschland Entwicklung GmbH, Schoenaicherstrasse 220, 71032 Boeblingen, Germany (leopoldj@de.ibm.com).* Mr. Leopold received a Dipl.-Inf. degree in computer science from the University of Applied Sciences, Konstanz, Germany. He is a member of the System z9 FCP channel firmware development team.