Dino Quintero
Alex Abderrazag
Shawn Bodily
Daniel J. Martin-Corben
Reshma Prathap
Kulwinder Singh
Ashraf Ali Thajudeen
William Nespoli Zanatta

IBM PowerHA SystemMirror for AIX Best Practices

Introduction

IBM® PowerHA® SystemMirror® for AIX® (formerly IBM HACMP™) was first available in 1991 and is now in its 24th release, with over 20,000 PowerHA clusters in production, worldwide. IBM PowerHA SystemMirror is recognized as a robust, mature high availability solution. PowerHA supports a wide variety of configurations, and offers a great deal of flexibility to the cluster administrator. With this flexibility comes the responsibility to make wise choices because many cluster configurations are available that work regarding the cluster passing verification and being brought online, but those configurations are not ideal in terms of providing availability.

This IBM Redpaper™ publication[1] describes choices that the cluster designer can make, and suggests the alternatives that can achieve the highest level of availability.

This paper discusses the following topics:

- Designing high availability
- Cluster components
- Testing
- Maintenance
- Monitoring
- PowerHA in a virtualized world
- Summary

[1] This document applies to PowerHA 7.1.3 SP1 running under AIX 7.1.3 TL1.

Designing high availability

A fundamental goal of a successful cluster design is the elimination of single points of failure (SPOF).

A high availability solution helps ensure that the failure of any component of the solution, whether it is hardware, software, or system management, does not cause the application and its data to be inaccessible to the user community. This solution is achieved through the elimination or masking of both planned and unplanned downtime. High availability solutions help eliminate single points of failure through appropriate design, planning, selection of hardware, configuration of software, and carefully controlled change management discipline.

To be highly available, a cluster must have no single point of failure. Although the principle of *no single point of failure* is accepted, it is sometimes inadvertently or deliberately violated. It is inadvertently violated when the cluster designer does not appreciate the consequences of the failure of a specific component. It is deliberately violated when the cluster designer chooses not to put redundant hardware in the cluster. The most common instance is when the cluster nodes that are chosen do not have enough I/O slots to support redundant adapters. This choice is often made to reduce the price of a cluster, and is generally a false economy; the resulting cluster is still more expensive than a single node, but has no better availability.

Plan a cluster carefully so that every cluster element has a backup (some say two of everything). A preferred practice is to use either paper or online planning worksheets to do this planning, and save them as part of the on-going documentation of the system. Table 1 lists typical SPOFs within a cluster.

Base the cluster design decisions on whether the cluster designs contribute to availability (that is, eliminate an SPOF) or detract from availability (gratuitously complex).

Table 1 Eliminating SPOFs

Cluster object	Eliminated as a single point of failure by these methods
Node	Use multiple nodes.
Power source	Use multiple circuits or uninterruptible power supplies (UPS).
Network adapter	Use redundant network adapters.
Network	Use multiple networks to connect nodes.
TCP/IP subsystem	Use non-IP networks to connect adjoining nodes and clients.
Disk adapter	Use redundant disk adapter or multipath hardware.
Disk	Use multiple disks with mirroring or RAID.
Application	Add a node for takeover; configure application monitor.
Administrator	Add backup or very detailed operation guide.
Site	Add an additional site.

Risk analysis

Sometimes in reality, eliminating all SPOFs within a cluster is not feasible. Examples might include network and site:

- If the network as a SPOF must be eliminated, then the cluster requires at least two networks. Unfortunately, this eliminates only the network directly connected to the cluster as an SPOF. It is not unusual for the users to be located some number of hops away from the cluster. Each of these hops involves routers, switches, and cabling, and each typically represents another SPOF. Truly eliminating the network as a SPOF can become a massive undertaking.
- Eliminating the site as a SPOF depends on distance and the corporate disaster recovery strategy. Generally, this involves using PowerHA SystemMirror Enterprise Edition. However, if the sites can be covered by a common storage area network, for example buildings within a 2 km radius, then cross-site Logical Volume Manager (LVM) mirroring function as described in the *PowerHA Administration Guide* is most appropriate, providing the best performance at no additional expense. If the sites are within the range of Peer-to-Peer Remote Copy (PPRC) (roughly, 100 km) and compatible IBM ESS, DS, SVC storage systems are used, then one of the PowerHA SystemMirror Enterprise Edition PPRC technologies is appropriate. Otherwise, consider PowerHA SystemMirror Global Logical Volume Manager (GLVM). For more information, see *IBM PowerHA Cookbook for AIX Updates*, SG24-7739.

Risk analysis techniques can be used to determine the SPOFs that simply must be handled and SPOFs that can be tolerated, as in this example:

- Study the current environment. Is the server room on a properly sized UPS, but no disk mirroring occurs today?
- Perform requirements analysis. How much availability is required and what is the acceptable likelihood of a long outage?
- Hypothesize all possible vulnerabilities. What might go wrong?
- Identify and quantify risks. What is the cost estimate of a failure versus the probability that it occurs?
- Evaluate counter measures. What is required to reduce the risk or consequence to an acceptable level?
- Finally, make decisions, create a budget and design the cluster.

Cluster components

The following section describes preferred practices for important cluster components.

Nodes

PowerHA v7.1 supports clusters of up to 16 nodes, with any combination of active and standby nodes. Although a possibility is to have all nodes in the cluster running applications (a configuration referred to as *mutual takeover*), the most reliable and available clusters have at least one standby node: one node that is normally not running any applications, but is available to take them over if a failure occurs on an active node.

Also, be sure to attend to environmental considerations. Nodes should not have a common power supply, which can happen if they are placed in a single rack. Similarly, building a cluster

of nodes that are actually logical partitions (LPARs) with a single footprint is useful as a test cluster, but do not consider them for availability of production applications.

Choose nodes that have sufficient I/O slots to install redundant network and disk adapters. (twice as many slots as is required for single node operation). This naturally suggests avoiding processors with small numbers of slots. For high availability best practices, do not consider or plan to use a node unless it has redundant adapters. Blades are an outstanding example. And, just as every cluster resource should have a backup, the root volume group in each node should be mirrored, or be on a RAID device. Furthermore, PowerHA v7.1 added the rootvg system event, which monitors rootvg and can help invoke a fallover in the event of rootvg loss.

Also, choose nodes so that, when the production applications are run at peak load, sufficient CPU cycles and I/O bandwidth still exist to allow PowerHA to operate. The production application should be carefully benchmarked (preferable) or modeled (if benchmarking is not feasible) and nodes chosen so that they do not exceed 85% busy, even under the heaviest expected load.

Note: Size the takeover node to accommodate all possible workloads: if a single standby is backing up multiple primaries, it must be capable of servicing multiple workloads.

On hardware that supports dynamic LPAR operations, PowerHA can be configured to allocate processors and memory to a takeover node before applications are started. However, these resources must actually be available, or acquirable through Capacity Upgrade on Demand (CUoD). Understand and plan for the worst case situation where, for example, all the applications are on a single node.

Networks

PowerHA is a network-centric application. PowerHA networks not only provide client access to the applications but are used to detect and diagnose node, network, and adapter failures. To do this, PowerHA uses these methods, which sends heartbeats over *all* defined networks:

- Before PowerHA v7: Reliable Scalable Cluster Technology (RSCT)
- PowerHA v7 and later: Cluster Aware AIX (CAA)

By gathering heartbeat information on multiple nodes, PowerHA can determine what type of failure occurred and initiate the appropriate recovery action. Being able to distinguish between certain failures, for example the failure of a network and the failure of a node, requires a second network. Although this additional network can be “IP based,” it is possible that the entire IP subsystem can fail within a given node. Therefore, in addition there should be at least one, ideally two, non-IP networks. Failure to implement a non-IP network can potentially lead to a partitioned cluster, sometimes referred to as the *split brain syndrome*. This situation can occur if the IP network between nodes becomes severed or in some cases congested. Because each node is still alive, PowerHA concludes the other nodes are down and initiates a takeover. After takeover, one or more applications might be running simultaneously on both nodes. If the shared disks are also online to both nodes, the result can lead to data divergence (massive data corruption). This is a situation that must be avoided, at all costs.

Starting in PowerHA v7 with the use of CAA, the new cluster repository disk automatically provides a form of non-IP heartbeating. Another option is to use SAN heartbeat, which is commonly referred to as *sancomm* or by the device name it uses called *sfwcomm*. Using sancomm requires SAN adapters that support *target mode* and zoning the adapters together so they can communicate with each other.

Important network best practices for high availability are as follows:

- Failure detection is possible only if at least two physical adapters per node are in the same physical network or VLAN. Be extremely careful when you make subsequent changes to the networks, with regards to IP addresses, subnetmasks, intelligent switch port settings and VLANs.
- The more unique types, both IP and non-IP, of networks the less likely of ever reporting a false-node-down failure.
- Where possible, use EtherChannel, Shared Ethernet Adapters (SEA), or both, through the Virtual I/O Server (VIOS) with PowerHA to aid availability.

  Note: PowerHA sees EtherChannel configurations as single adapter networks. To aid problem determination, configure the `netmon.cf` file to allow ICMP echo requests to be sent to other interfaces outside of the cluster. See the PowerHA administration web page for further details:

  `http://www-01.ibm.com/support/knowledgecenter/SSPHQG_7.1.0/com.ibm.powerha.admngd/ha_admin_kickoff.htm`

- When you use multiple adapters per network, each adapter needs an IP address in a different subnet, using the same subnet mask.
- Currently, PowerHA supports IPv6 and Ethernet only.
- Ensure you have in place the correct network configuration rules for the cluster with regards to EtherChannel, Virtual adapter support, service, and persistent addressing. For more information, see the PowerHA planning web page:

  `http://www-01.ibm.com/support/knowledgecenter/SSPHQG_7.1.0/com.ibm.powerha.plangd/ha_plan.htm`

- Name resolution is essential for PowerHA. External resolvers are deactivated under certain event processing conditions. Avoid problems by configuring `/etc/netsvc.conf` and NSORDER variable in `/etc/environment` to ensure that the host command checks the local `/etc/hosts` file first.
- Read the release notes that are stored in `/usr/es/sbin/cluster/release_notes`. Watch for new and enhanced features, such as collocation rules, persistent addressing and fast failure detection.
- Configure persistent IP labels to each node. These IP addresses are available at AIX boot time and PowerHA strives to keep them highly available. They are useful for remote administration, monitoring, and secure node-to-node communications. Consider implementing a host-to-host IPsec tunnel between persistent labels between nodes. This can ensure that sensitive data, such as passwords, are not sent unencrypted across the network, for example when using the C-SPOC option to change a user password.
- If you have several virtual clusters split across frames, ensure boot subnet addresses are unique per cluster. This minimizes problems with `netmon` reporting the network is up when indeed the physical network outside the cluster might be down.

Adapters

As stated previously, each network defined to PowerHA should have at least two adapters per node. Although it is possible to build a cluster with fewer, the reaction to adapter failures is more severe; the resource group must be moved to another node. AIX provides support for both EtherChannel and Shared Ethernet Adapters. This often allows the cluster node to logically have defined one adapter interface per network. This reduces the number of IP

addresses required, allows the boot IP address and service IP to be on the same subnet, and can result in not needing to define a persistent addresses.

Many IBM Power Systems™ servers contain built-in virtual Ethernet adapters. These historically have been known as Integrated Virtual Ethernet (IVE) or Host Ethernet Adapters (HEA). Some newer systems now contain Single Root I/O Virtualization (SRIOV) adapters. Most of these adapters provide multiple ports. One port on such an adapter should not be used to back up another port on that adapter, because the adapter card is a common point of failure. The same is often true of the built-in Ethernet adapters; in most IBM Power Systems servers, ports have a common adapter. When the built-in Ethernet adapter can be used, a preferred practice is to provide an extra adapter in the node, with the two backing up each other. However, be aware that, when using these specific types of adapters, in many cases, Live Partition Mobility might be unable to be used.

Also be aware of network detection settings for the cluster and consider tuning these values. These values apply to *all* networks. However, be careful when you use custom settings, because setting these values too low can lead to undesirable results, like false takeovers. These settings can be viewed and modified by using either the **clmgr** command or **smitty sysmirror**.

Applications

The most important part of making an application run well in a PowerHA cluster is understanding the application's requirements. This is particularly important when designing the *resource group* policy behavior and dependencies. For high availability to be achieved, the application must be able to stop and start cleanly and not explicitly prompt for interactive input. Some applications tend to bond to a particular operating system characteristic such as a uname, serial number, or IP address. In most situations, these problems can be overcome. The vast majority of commercial software products that run under AIX are suited to be clustered with PowerHA.

Application data location

Where should application binaries and configuration data reside? There are many arguments to this discussion. Generally, keep all the application binaries and data where possible on the shared disk, because forgetting to update it on all cluster nodes when it changes is easy. This can prevent the application from starting or working correctly when it is run on a backup node. However, the correct answer is not firm. Although many application vendors have suggestions for how to set up the applications in a cluster, these are recommendations. Just when it seems to be clear cut as to how to implement an application, someone thinks of a new set of circumstances. Here are several guidelines:

- If the application is packaged in LPP format, it is usually installed on the local file systems in rootvg. This behavior can be overcome by storing the install packages to disk by using the **bffcreate** command and then restoring them with the preview option. This action shows the installation paths, and then symbolic links can be created before installation, which points to the shared storage area.
- If the application is to be used on multiple nodes with different data or configurations, then the application and configuration data are probably on local disks and the data sets on shared disk with application scripts, altering the configuration files during fallover.
- Also, remember the PowerHA file collections facility can be used to keep the relevant configuration files in sync across the cluster. This is useful for applications that are installed locally.

Start and stop scripts

Application *start* scripts should not assume the status of the environment. Intelligent programming must correct any irregular conditions that might occur. The cluster manager spawns theses scripts in a separate job in the background and carries on processing. Some tasks a start script should perform are as follows:

1. Check that the application is not currently running. This is important because resource groups can be placed into an unmanaged state (forced-down action, in previous versions). Using the default startup options, PowerHA will rerun the application start script, which might cause problems if the application is running. A simple and effective solution is to check the state of the application on startup. If the application is found to be running, end the start script with `exit 0`.
2. Verify the environment. Are all the disks, file systems, and IP labels available?
3. If different commands are to be run on different nodes, store the executing HOSTNAME to a variable.
4. Check the state of the data. Does it require recovery? Always assume the data is in an unknown state since the conditions that occurred to cause the takeover cannot be assumed.
5. Do prerequisite services exist that must be running? Is it feasible to start all prerequisite services from within the start script? Is there an inter-resource group dependency or resource group sequencing that can guarantee that the previous resource group started correctly? PowerHA can implement checks on resource group dependencies including collocation rules.
6. When the environment looks correct, start the application. If the environment is not correct and error recovery procedures cannot fix the problem, ensure that adequate alerts (email, SMS, SMTP traps, and others) are sent over the network to the appropriate support administrators.

The *stop* scripts differ from start scripts in that most applications have a documented start-up routine and not necessarily a stop routine. The assumption is that once the application is started, why stop it? Relying on a failure of a node to stop an application will be effective, but to use some of the more advanced features of PowerHA, the requirement exists to stop an application cleanly. Avoid the following issues, among others:

- Be sure to terminate any child or spawned processes that might be using disk resources. Consider implementing child resource groups.
- Verify that the application is stopped to the point that the file system is free to be unmounted. The **`fuser`** command can verify that the file system is free.
- In some cases, it might be necessary to double-check that the application vendor's stop script actually stopped all the processes; sometimes it might be necessary to terminate some processes by force. Clearly the goal is to return the machine to the state it was in before the application start script was run.
- Failure to exit the stop script with a zero return code will stop cluster processing.

> **Note:** This is not the case with start scripts when using background startup option.

Remember, most vendor stop/starts scripts are not designed to be cluster proof. A useful tip is to have stop and start script verbosely output using the same format to the `/tmp/hacmp.out` file. This can be achieved by including the following line in the header of the script:

```
set -x && PS4="${0##*/}"'[$LINENO] '
```

Application monitoring

With PowerHA, you can monitor the state of an application. Although optional, implementation is highly suggested. This mechanism provides for self-healing clusters. To ensure that event processing does not hang because of failures in the user supplied script and to prevent hold-up during event processing, PowerHA has always started the application in the background. This approach has disadvantages:

- There is no wait or error checking.
- In a multitiered environment, there is no easy way to ensure that applications of higher tiers have been started.

Application monitoring can either check for process death or run a user-supplied custom monitor method during the start-up or continued running of the application. The latter is particularly useful when the application provides some form of transaction processing: a monitor can run a null transaction to ensure that the application is functional. The preferred practice for applications is to have both process death and user-supplied application monitors in place.

More information about application monitoring is in the draft of *PowerHA SystemMirror for AIX Cookbook Update*, SG24-7739-01, which has an expected publish date of Q4 of 2014.

Do not forget to test the monitoring, and start, restart, and stop methods carefully. Poor start, stop, and monitor scripts can cause cluster problems, not only in maintaining application availability but avoiding data corruption.

Behavior: By having monitoring scripts exit with non-zero return codes when the application has not failed, in conjunction with poor start/stop scripts, can result in undesirable behavior (for example, data corruption). The application is down and is also in need of emergency repair that might involve restoring data from backup.

In addition, PowerHA also supplies a number of tools and utilities to help in customization efforts like pre- and post-event scripts and user-defined resources. Be careful to use only those for which PowerHA also supplies a man page (**`lslpp -f cluster.man.en_US.es.data`**) because those are the only ones for which upward compatibility is guaranteed. A good example for this use is application provisioning.

Application provisioning

PowerHA can drive Dynamic LPAR and some Capacity on Demand (CoD) operations to ensure that adequate processing and memory are available for the one or more applications upon start-up. This is shown in Figure 1 on page 9. Also see the following web page for information about supported Capacity Upgrade on Demand (CUoD) types:

`http://www-01.ibm.com/support/knowledgecenter/SSPHQG_7.1.0/com.ibm.powerha.admngd/ha_admin_types_cuod_licenses.htm`

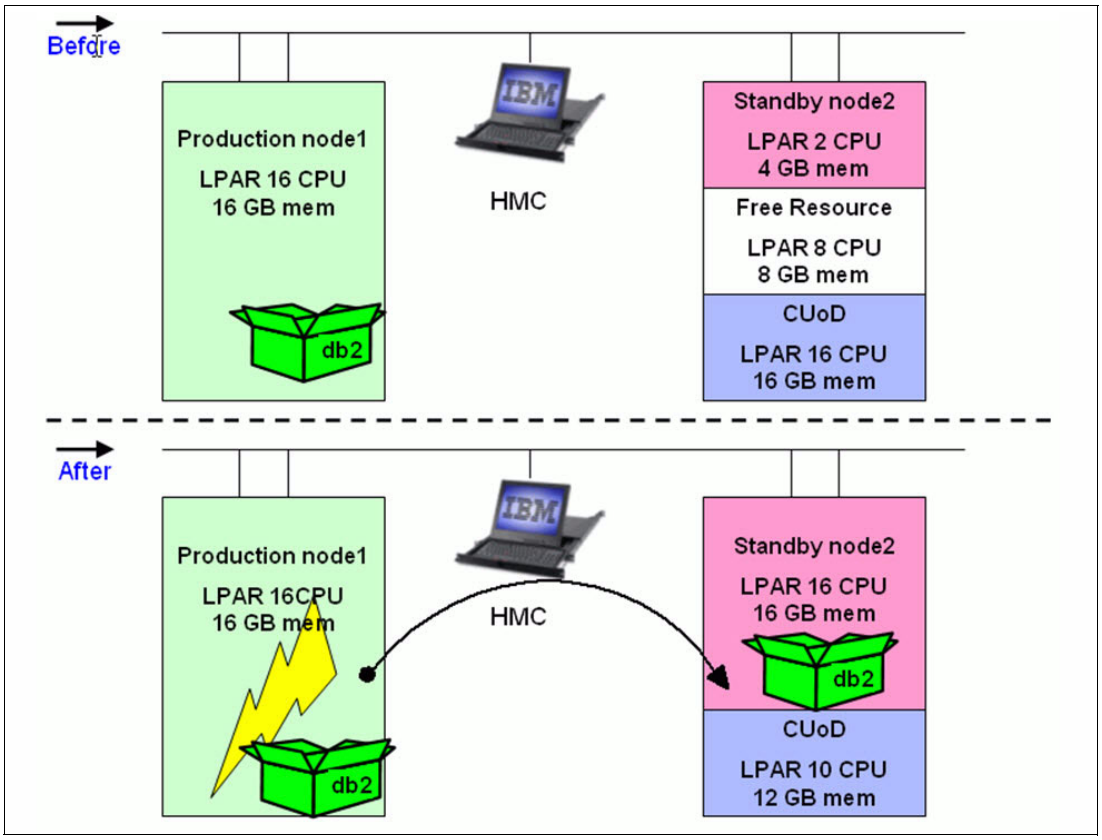

Figure 1 Application provisioning example

This process can be driven by using PowerHA SMIT panels. However, consider the following information about this approach:

- The CoD activation key must be entered manually *before* any PowerHA dynamic logical partitioning (DLPAR) event.
- The LPAR name must be the AIX operating system host name, which must be the PowerHA node name.
- Large memory moves are actioned in one operation. This can be time-consuming and delay event processing.
- The LPAR host name must be resolvable at the Hardware Management Console (HMC).
- If the acquisition or release fails, the operation is not repeated on another HMC if defined.

More detail about using and configuring this option is in the draft of *PowerHA SystemMirror for AIX Cookbook Update*, SG24-7739-01, which has an expected publish date of Q4 of 2014.

Testing

Although this statement sounds simplistic, the most important step in testing is to actually do it.

A cluster should be thoroughly tested before initial production (and after **`clverify`** command runs without errors or warnings). This means that every cluster node and every interface that PowerHA uses must be stopped and started again to validate that PowerHA responds as expected. Be sure to perform the same level of testing after each change to the cluster.

PowerHA offers a cluster test tool that can be run on a cluster before it is put into production. This will verify that the applications are brought back online after node, network, and adapter failures. Run the test tool as part of any comprehensive cluster test effort.

More information about the cluster test tool is in the draft of *PowerHA SystemMirror for AIX Cookbook Update*, SG24-7739-01, which has an expected publish date of Q4 of 2014.

In addition, plan for regular testing. A common safety recommendation is to test home smoke detectors twice a year; the switch to and from Daylight Saving Time are commonly when to test. Similarly, if the enterprise can afford to schedule it, node fallover and fallback tests should be scheduled bi-annually. These tests at least indicate whether any problems have crept in, and allow for correction before the cluster fails in production.

On a more regular basis, run the **clverify** command. Seriously consider errors and warning messages, and correct those problems at the first opportunity. The **clverify** command runs automatically daily at 00:00 hours. Part of the practice for administrators should be to routinely check the logs daily, and react to any warnings or errors.

Maintenance

Even the most carefully planned and configured cluster might experienced problems if it is not well maintained. A large part of best practice for a PowerHA cluster is associated with maintaining the initial working state of the cluster through hardware and software changes.

Before any change to a cluster node, take a PowerHA snapshot. If the change involves installing a PowerHA, AIX, or other software fix, also take a **mksysb** backup, use **multibos**, or **alt_disk_install** (`alt_disk_copy`, `alt_clone`, `alt_disk_mksysb` are useful options). Also *apply* fixes and updates instead of *commit.* In this way, removing fixes, if necessary, is easier.

On successful completion of the change, use SMIT to display the cluster configuration, print, and save the `smit.log` file. The **clmgr** facility can also be used to generate an HTML report of the cluster configuration in PowerHA v7.1.3.

All mission-critical high availability cluster enterprises should, as a preferred practice, maintain a test cluster identical to the production ones. Thoroughly test all changes to applications, cluster configuration, or software on the test cluster before putting them on the production clusters. To at least partially automate this effort, use the PowerHA cluster test tool.

Change control is important in a PowerHA cluster. In some organizations, databases, networks, and clusters are administered by separate individuals or groups. When any group plans maintenance on a cluster node, it should be planned and coordinated among all parties. All should be aware of the changes being made to avoid introducing problems. Organizational policy must preclude “unilateral” changes to a cluster node. In addition, change control in a PowerHA cluster must include a goal of having all cluster nodes at the same level. Upgrading only the node running the application is insufficient (and not recommended). Develop a process that encompasses the following set of questions:

- Is the change necessary?
- How urgent is the change?
- How important is the change? (This is not the same as urgent.)
- What impact does the change have on other aspects of the cluster?
- What is the impact if the change is not allowed to occur?
- Are all steps needed to implement the change clearly understood and documented?
- How will the change be tested?

- What is the plan for backing out the change if necessary?
- Is the appropriate expertise available if problems develop?
- When is the change scheduled?
- Were the users notified?
- Does the maintenance period include sufficient time for a full set of backups before the change and sufficient time for a full restore afterwards should the change fail testing?

This process should include an electronic form, which requires appropriate signoffs before the change can go ahead. Every change, even the minor ones, must follow the process. The notion that a change, even a small change, might be permitted (or sneaked through) without following the process must not be permitted.

To this end, the preferred practice is to use the PowerHA C-SPOC facility, or C-SPOC command line equivalent, where possible for any change. Especially with regards to shared volume groups. If the installation uses AIX password control on the cluster nodes (as opposed to NIS or LDAP), C-SPOC should also be used for any changes to users and groups. PowerHA will then ensure that the change is properly reflected to all cluster nodes.

More information about cluster maintenance and administration is in the draft of *PowerHA SystemMirror for AIX Cookbook Update*, SG24-7739-01, which has an expected publish date of Q4 of 2014.

Upgrading the cluster environment

OK, so you want to upgrade? Start by reading the upgrade chapter in the PowerHA installation documentation and make a detailed plan:

`http://ibm.co/1qkduDw`

Taking the time to review and plan thoroughly will save many “I forgot to do that” problems during and after the migration or upgrade process. Remember to check all version compatibilities between the levels of software and firmware, and, most important, the application software certification against the level of AIX and PowerHA. If you are not sure, check with IBM support or use the Fix Level Recommendation Tool (FLRT):

`http://www14.software.ibm.com/webapp/set2/flrt/home`

Do not attempt to upgrade AIX or PowerHA without first taking a backup and checking that it is restorable. In all cases, completing the process in a test environment before actually doing it for real is extremely useful. AIX facilities, such as **`alt_disk_copy`** and **`multibos`** for creating an alternative rootvg, which can be activated by rebooting, and are useful tools worth exploring and using.

Before attempting the upgrade, complete the following steps:

1. Check that cluster and application are stable and that the cluster can synchronize cleanly.
2. Take a cluster snapshot and save it to a temporary non-cluster directory (export `SNAPSHOTPATH=<some other directory>`).
3. Save event script customization files and user-supplied scripts to a temporary non-cluster directory. If you are unsure that any custom scripts are included, check by using **`odmget HACMPcustom`** command.
4. Check that the same level of cluster software (including program temporary fixes (PTFs)) are on all nodes before beginning a migration.
5. Ensure that the cluster software is committed (and not just applied).

Where possible, use the rolling migration method to ensure maximum availability. Effectively, cluster services are stopped one node at a time by using the takeover option (now referred to as move resource groups). The node or system is updated accordingly and cluster services are restarted. This operation is completed one node at a time until all nodes are at the same level and are operational.

Note: Although PowerHA will work with mixed levels of AIX or PowerHA in the cluster, the goal is to have all nodes at exactly the same levels of AIX, PowerHA, and application software. In addition, PowerHA prevents changes to the cluster configuration when mixed levels of PowerHA are present.

PowerHA service packs can now be applied using a *nondisruptive update* method. The process is actually identical to the rolling migration, however, resource groups are placed into an *unmanaged state* to ensure they remain available and performed one at a time from start to finish.

Note: During this state, the application (or applications) are not under the control of PowerHA (for example, not highly available). Using the default startup options, PowerHA relies on an application monitor to determine the application state and hence appropriate actions to take.

Alternatively, the entire cluster and applications can be gracefully shut down to update the cluster using either the *snapshot* or *offline* conversion methods. Historically, upgrading the cluster this way has resulted in fewer errors but requires a period of down time.

Tip: Demos are available for performing migrations; see the following web address:

`http://www.youtube.com/PowerHAguy`

More information about the migration process is in the draft of *PowerHA SystemMirror for AIX Cookbook Update*, SG24-7739-01, which has an expected publish date of Q4 of 2014.

Monitoring

In a clustered environment, gaining timely and accurate status information regarding the cluster topology and application resources is critical. Also critical is for application monitors to be configured for each application that is to be made highly available in the cluster[2]. Without application monitors, PowerHA has no mechanism to know whether your applications are actually available and performing as you expect.

PowerHA provides commands such as **`cldump`** and **`clstat`** for monitoring the status of the cluster.

[2] Application Monitoring is a feature of PowerHA and aides the cluster in determining whether the application is alive and well. Application Monitoring is beyond the scope of this chapter.

The SNMP protocol is the crux to obtaining the status of the cluster. The SNMP protocol is used by network management software and systems for monitoring network applications and devices for conditions that warrant administrative attention. SNMP protocol consists of a database, and a set of data objects. The set of data objects forms a Management Information Base (MIB). The standard SNMP agent is the **snmpd** daemon. An SNMP Multiplexing protocol (SMUX) subagent allows vendors to add more MIB information that is product-specific. The **clstrmgr** daemon in PowerHA acts as a SMUX subagent. The SMUX peer function, contained in **clstrmgrES** daemon, maintains cluster status information for the PowerHA MIB. When the clstrmgrES starts, it registers with the SNMP daemon, **snmpd**, and continually updates the MIB with cluster status information in real time. PowerHA implements a private MIB branch maintained through a SMUX peer subagent to SNMP contained in **clstrmgrES**, as shown in Figure 2.

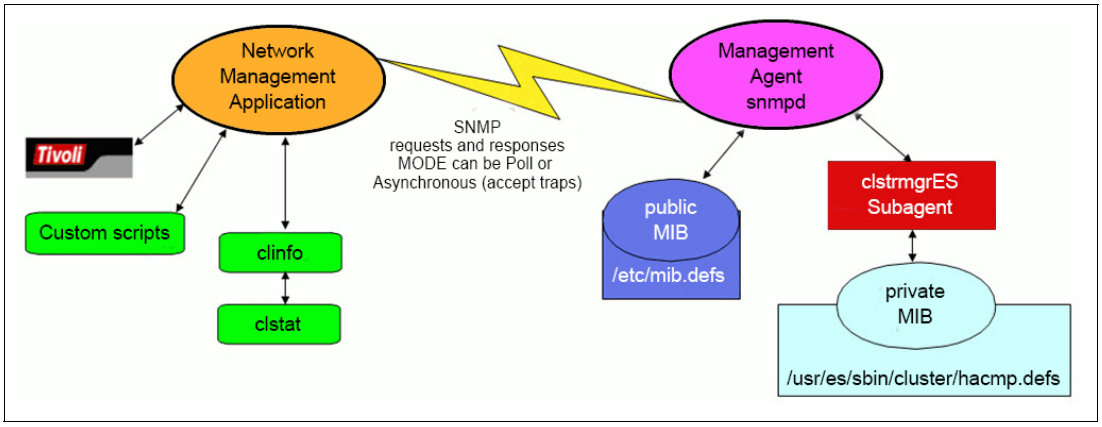

Figure 2 PowerHA private managed information base (MIB)

The **clinfo** daemon status facility has a few considerations and many users or administrators of PowerHA clusters implement custom monitoring scripts. This might seem complex but actually it is remarkably straight forward. The cluster SNMP MIB data can be pulled simply over a secure session by using the following command:

```
ssh $NODE snmpinfo -v -m dump -o /usr/es/sbin/cluster/hacmp.defs risc6000clsmuxpd
> $OUTFILE
```

PowerHA participates under the IBM Enterprise SNMP MIB (Figure 3):

ISO (1) → Identified Organization (3) → Department of Defense (6) → Internet (1) → Private (4) → Enterprise (1) → IBM (2) → IBM Agents (3) → AIX (1) → aixRISC6000 (2) → risc6000agents (1) → risc6000clsmuxpd (5)

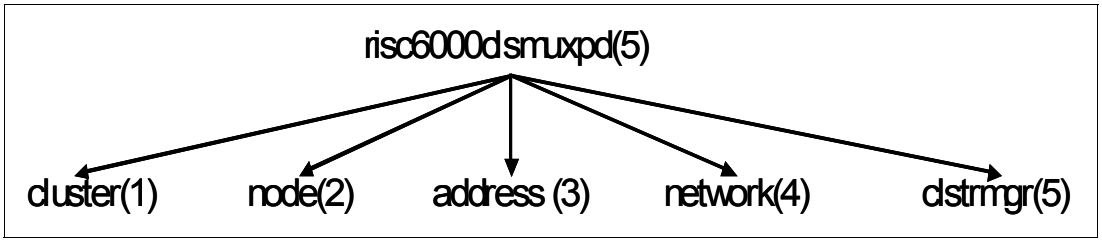

Figure 3 PowerHA cluster MIB structure

The resultant MIB for PowerHA **cluster** would be 1.3.6.1.4.1.2.3.1.2.1.5.1. The data held within this MIB can be pulled using the **snmpinfo** command as shown in Example 1 on page 14.

Example 1 snmpinfo command

```
# snmpinfo -v -m dump -o /usr/es/sbin/cluster/hacmp.defs cluster
clusterId.0 = 1120652512
clusterName.0 = "sapdemo71_cluster"
clusterConfiguration.0 = ""
clusterState.0 = 2
clusterPrimary.0 = 1
clusterLastChange.0 = 1386133818
clusterGmtOffset.0 = 21600
clusterSubState.0 = 32
clusterNodeName.0 = "mhoracle1"
clusterPrimaryNodeName.0 = "mhoracle1"
clusterNumNodes.0 = 2
clusterNodeId.0 = 1
clusterNumSites.0 = 0
```

Individual elements, for example the cluster state and cluster sub state, can be pulled as shown in Example 2.

Example 2 Showing the cluster state

```
# snmpinfo -v -o /usr/es/sbin/cluster/hacmp.defs ClusterState.0
clusterState.0 = 2

# snmpinfo -v -o /usr/es/sbin/cluster/hacmp.defs ClusterSubState.0
clusterSubState.0 = 32

Note: the -v translates the numbered MIB branch path to reable variable name.

# snmpinfo -o /usr/es/sbin/cluster/hacmp.defs ClusterState.0
1.3.6.1.4.1.2.3.1.2.1.5.1.4.0 = 2
```

Example 2 shows that the cluster has a state of 2 and a substate of 32. To determine the meaning of these values, see the `/usr/es/sbin/cluster/hacmp.my` file, which contains a description of each HACMP MIB variable (Example 3).

Example 3 Snapshot of the HACMP MIB definition file

```
clusterState    OBJECT-TYPE
                SYNTAX  INTEGER { up(2), down(4),
                                  unknown(8), notconfigured(256) }
                ACCESS  read-only
                STATUS  mandatory
                DESCRIPTION
                        "The cluster status"

clusterSubState OBJECT-TYPE
                SYNTAX  INTEGER { unstable(16), error(64),
                                  stable(32), unknown(8), reconfig(128),
                                  notconfigured(256), notsynced(512) }
                ACCESS  read-only
                STATUS  mandatory
                DESCRIPTION
                        "The cluster substate"
```

We can conclude from Example 3 that the cluster status is *up* and *stable*. This is the mechanism that **clinfo**/**clstat** uses to display the cluster status.

The **clstat** utility uses clinfo library routines (through **clinfo** daemon) to display all node, interface, and resource group information for a selected cluster. The **cldump** does likewise as a one-off command by interrogating the private MIB directly within the cluster node. Both are solely reliant on the SNMP protocol and rely on the mechanism described previously.

Graphical monitoring can also be performed from IBM Systems Director by using the PowerHA SystemMirror plug-in.

More information and options about cluster monitoring are in the draft of *PowerHA SystemMirror for AIX Cookbook Update*, SG24-7739-01, which has an expected publish date of Q4 of 2014.

PowerHA in a virtualized world

PowerHA works with virtual devices, however some restrictions apply when using virtual Ethernet or virtual disk access. Creating a cluster in a virtualized environment will add new SPOFs, which must be considered. PowerHA nodes inside the same physical footprint (frame) must be avoided if high availability is to be achieved; consider this configuration only for test environments. To eliminate the additional SPOFs in a virtual cluster, implement the use of a second VIOS in each frame with the Virtual I/O Client (VIOC) LPARs that are within different frames, ideally some distance apart.

Redundancy for disk access can be achieved through LVM mirroring, RAID and/or Multi-Path I/O (MPIO). LVM mirroring is most suited to eliminate the VIOC rootvg as a SPOF, as shown in Figure 4. The root volume group can be mirrored using standard AIX practices. In the event of VIOS failure, the LPAR will see stale partitions and the volume group must be resynchronized by using **syncvg**. This procedure can also use logical volumes as backing storage to maximize flexibility. For test environments, whereby each VIOC is located in the same frame, LVM mirroring can be used for datavgs also.

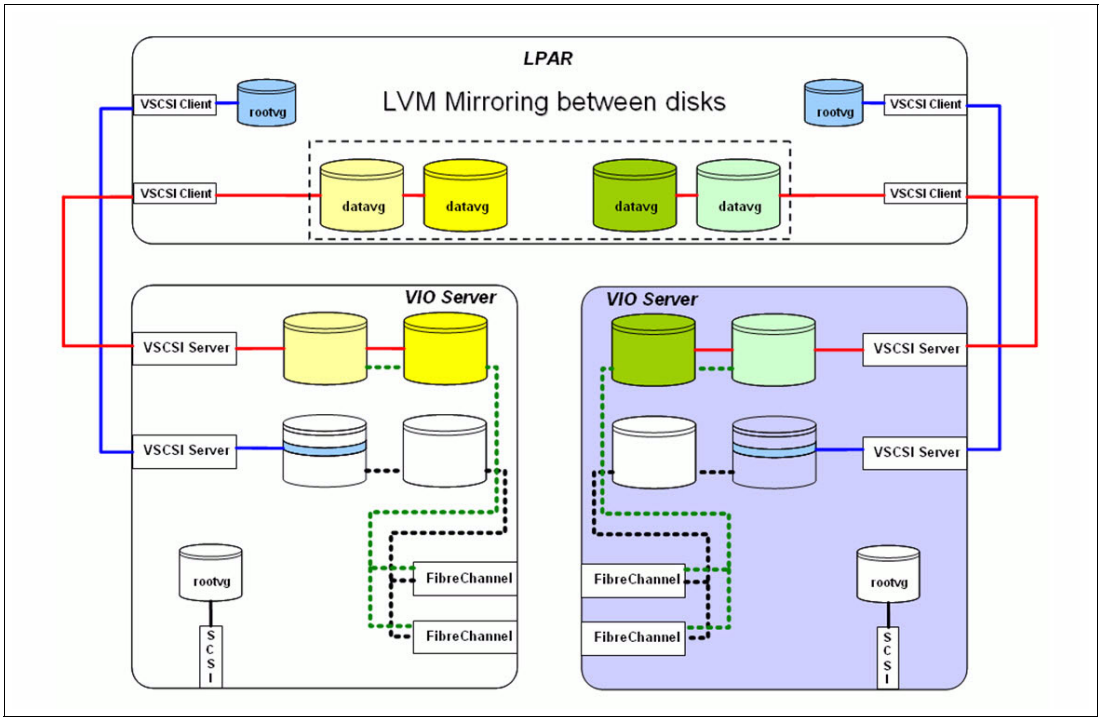

Figure 4 Redundancy using LVM Mirroring

For shared data volume groups, deploy the MPIO method. Figure 5. A LUN is mapped to both VIOS in the SAN. From both Virtual I/O Servers, the LUN is mapped again to the same VIOC. The VIOC LPAR will correctly identify the disk as an MPIO-capable device and create one hdisk device with two paths. The configuration is then duplicated on the backup frame or node. Currently, the virtual storage devices work only in failover mode, other modes are not yet supported. All devices accessed through a VIOS must support a `no_reserve` attribute. If the device driver is not able to “ignore” the reservation, the device can not be mapped to a second VIOS. Currently, the reservation held by a VIOS cannot be broken by PowerHA, hence only devices that will not be reserved on open are supported. Therefore, PowerHA requires the use of enhanced concurrent mode volume groups (ECVGs). The use of ECVGs is a general requirement starting with PowerHA v7.1.0.

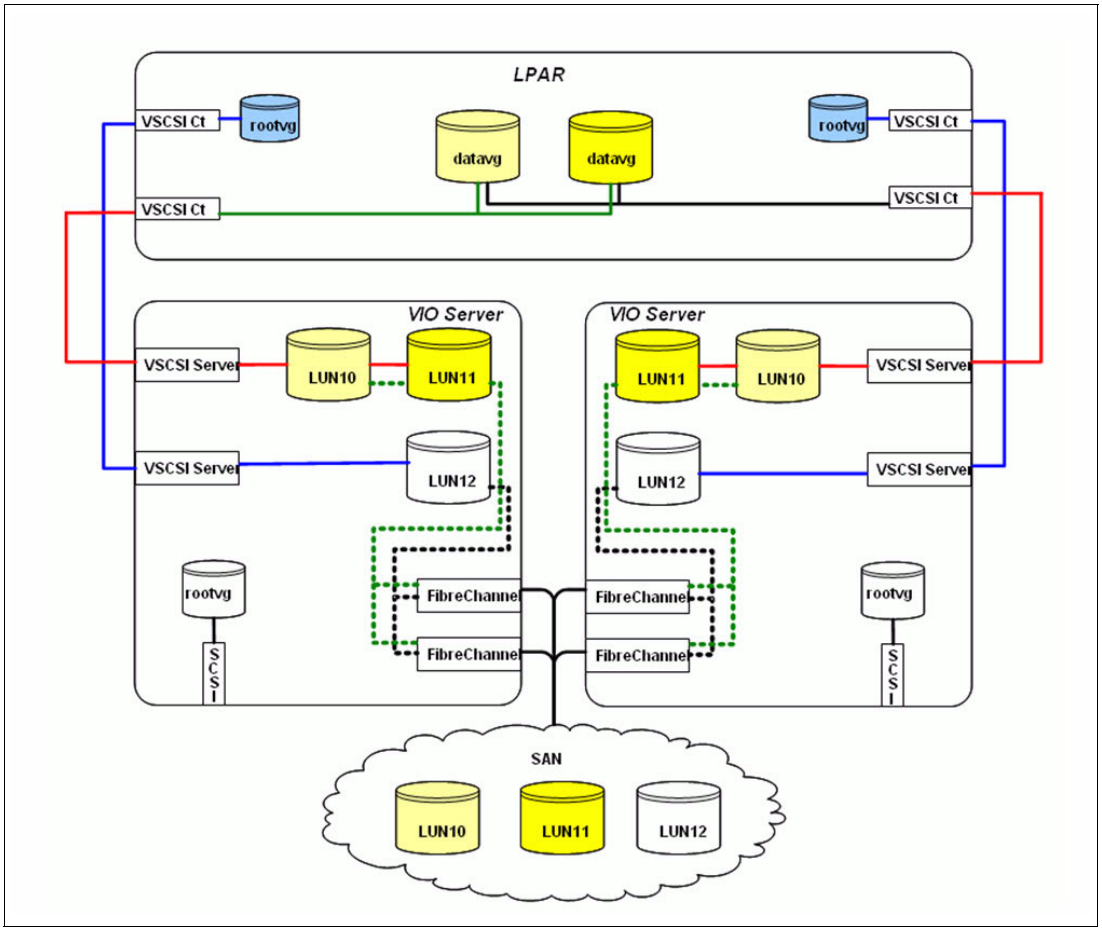

Figure 5 Redundancy using MPIO

In a virtualized networking environment, a VIOS is needed for access to the outside world via a layer-2 based Ethernet bridge, which is referred to as a Shared Ethernet Adapter (SEA). Now, the physical network devices along with the SEA are the new SPOFs. How are these SPOFs eliminated? Again through the use of a second VIOS. EtherChannel technology from within the VIOS can use used to eliminate both the network adapters and switch as a SPOF. To eliminate the VIOS as a SPOF, two choices are available:

- EtherChannel (configured in backup mode *only, no aggregation*) in the VIOC. See Figure 6 on page 17.
- SEA failover through the hypervisor. See Figure 7 on page 18.

Both methods have advantages and disadvantages. However, SEA failover is generally considered the preferred practice because it provides the use of Virtual LAN ID (VID) tags and keeps the client configuration cleaner.

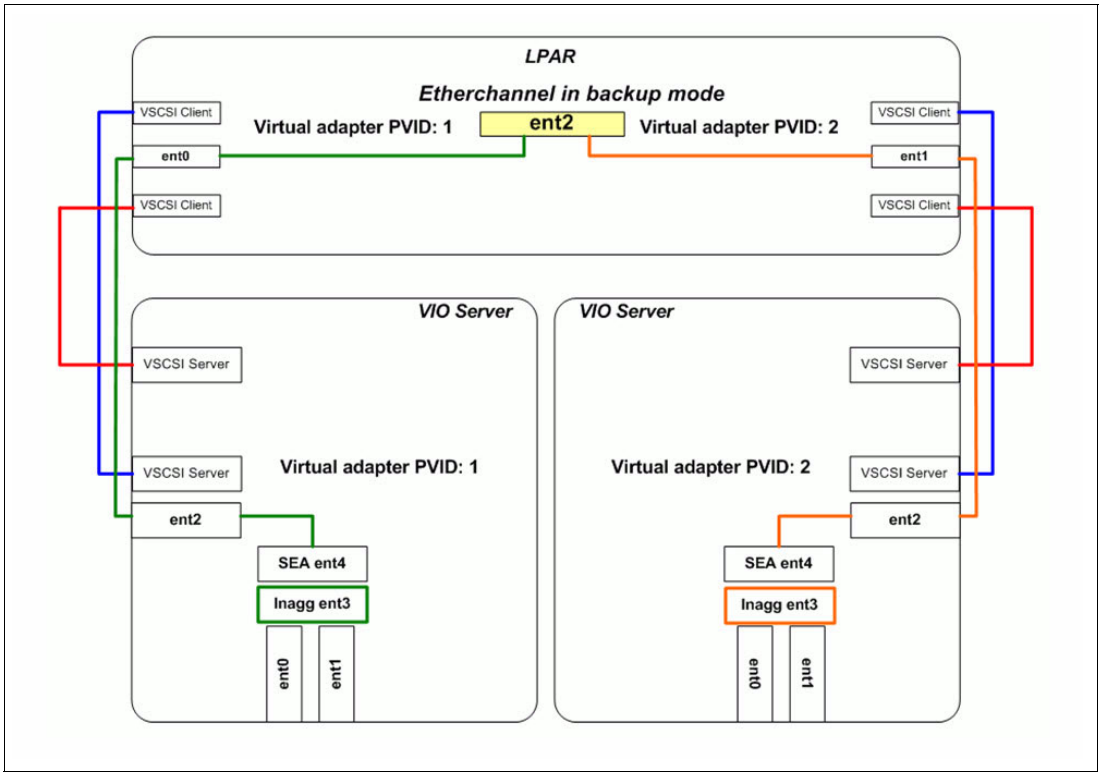

Figure 6 EtherChannel in backup mode

From the client perspective only a single virtual adapter is required. However having a second virtual adapter will not eliminate a SPOF because the adapter is not real. The SPOF is the hypervisor. Generally, single interface networks are not a preferred practice because this limits the error detection capabilities of PowerHA. In this case, it cannot be avoided so to help with further analysis, add external IP addresses to the `netmon.cf` file. In addition, at least two physical adapters per SEA should be used in the VIOS in an EtherChannel configuration. Adapters in this channel can also form an aggregate, but remember that most vendors require adapters that form an aggregate to share the same backplane (a SPOF, so do not forget to define a backup adapter). An exception to this rule is Nortel's Split Multi-Link Trunking. Depending on your environment, this technology might be worth investigating.

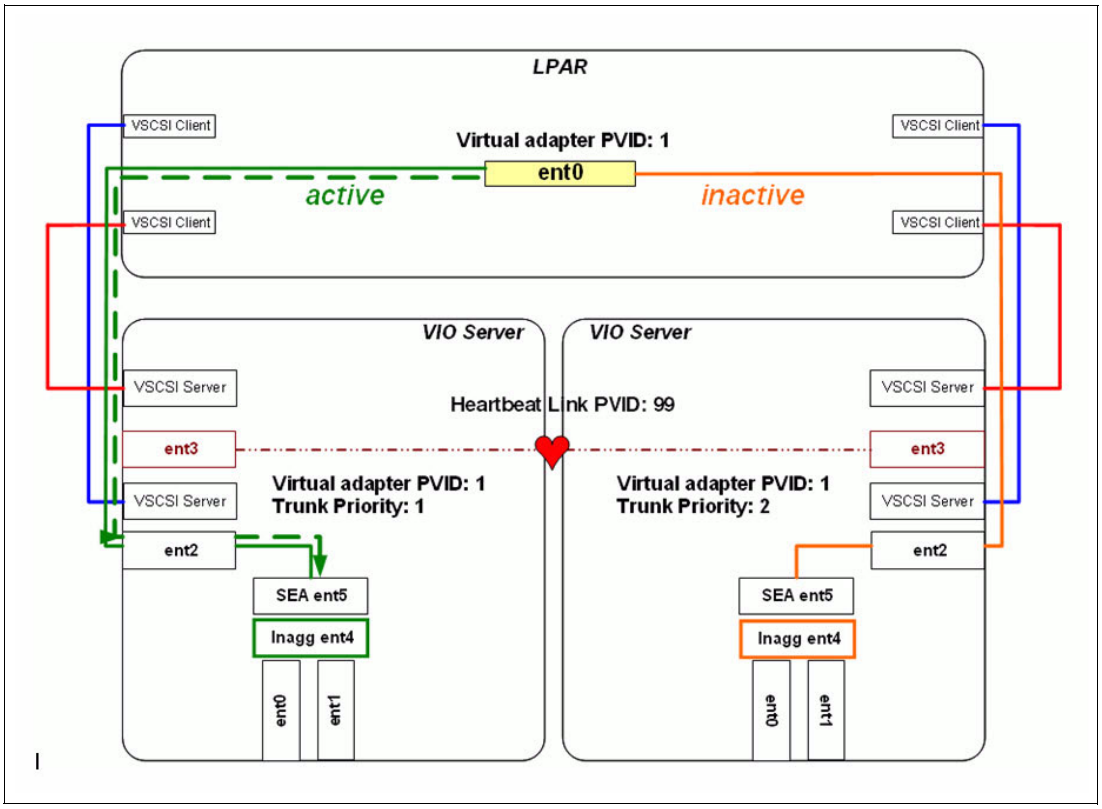

Figure 7 SEA failover

Finally, you see a view of the big picture. Be methodical in your planning. As Figure 8 on page 19 shows, even a simple cluster design can soon become rather complex.

More information about implementing PowerHA in a virtualized environment, including the use of NPIV, is in the draft of *PowerHA SystemMirror for AIX Cookbook Update*, SG24-7739-01, which has an expected publish date of Q4 of 2014.

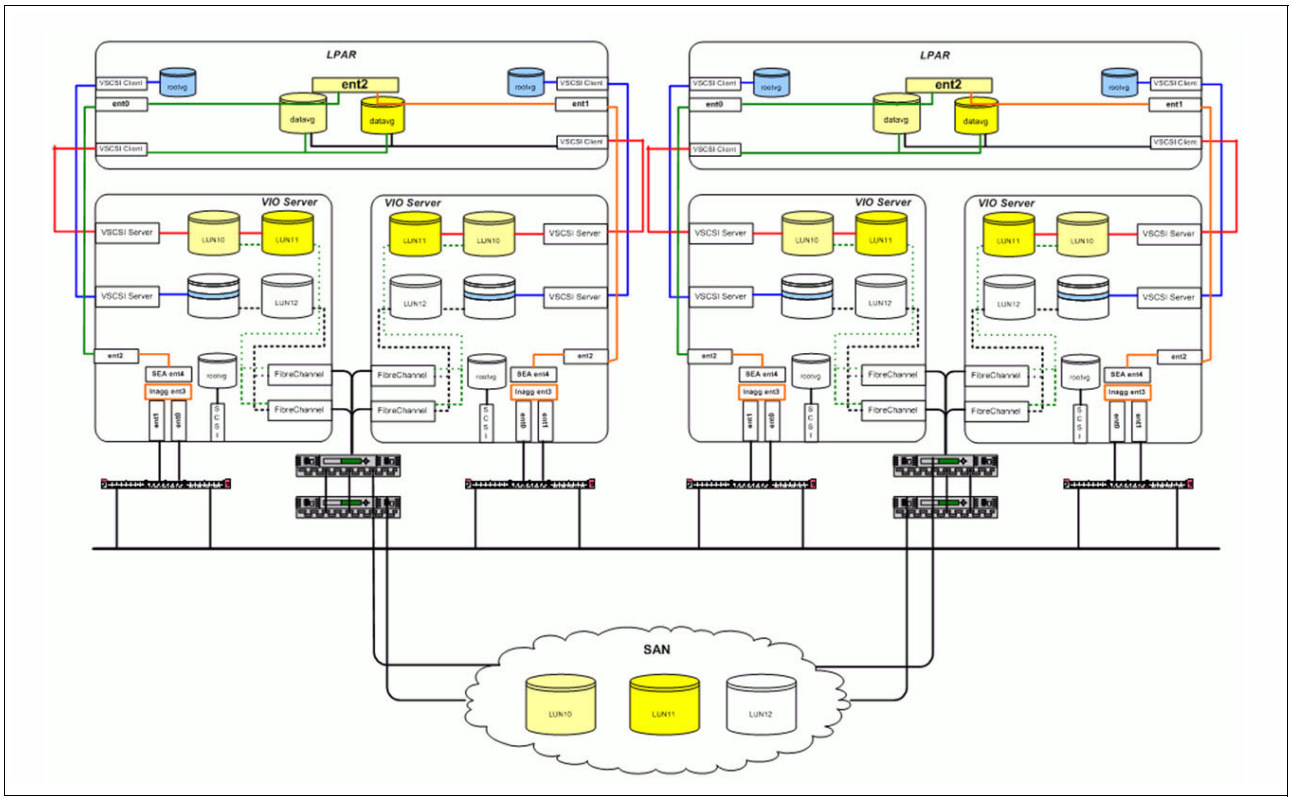

Figure 8 A PowerHA cluster in a virtualized world

Maintenance of the VIOS partition: Applying updates

The VIOS must be updated in isolation, that is, with no client access. A simple way of achieving this is to start by creating a new profile for the VIOS by copying the existing one. Then, delete all virtual devices from the profile and reactivate the VIOS using the new profile. This ensures that no client partition can access any devices and that the VIOS is ready for maintenance.

Before restarting the VIOS, manual failover from the client must be performed so all disk access and networking goes through the alternate VIOS. The steps to accomplish this are as follows:

1. For MPIO storage, disable the activate path by using the following command:

   ```
   chpath -l hdiskX -p vscsiX -s disable
   ```

2. For LVM mirrored disks, set the virtual SCSI target devices to a defined state in the VIOS partition.
3. Initiate the SEA failover from the active VIOS by using the following command:

   ```
   chdev -dev entX -attr ha_mode=auto
   ```

4. For EtherChannel in the VIOC, initiate a force failover by using the following command:

   ```
   smitty etherchannel
   ```

After the update is applied, reboot the VIOS. The client is then redirected to the newly updated VIOS and the same procedure is followed on the alternative VIOS. An important factor is to be sure that each VIOS used has the same code level.

Workload partitions

Workload partitions (WPARs) are software-created virtualized operating system environments within a single instance of the AIX operating system. WPARs secure and isolate the environment for the processes and signals that are used by enterprise applications.

WPARs types are application WPARs or system WPARs. System WPARs are autonomous virtual system environments with their own private file systems, users and groups, login, network space, and administrative domain.

By default, a system WPAR shares the two file systems named `/usr` and `/opt` from the global environment by using read-only namefs mounts. You can configure WPARs to have a non-shared, writable `/usr` file system and `/opt` file system. The WPARs are also called private.

For more information about IBM AIX WPARs, see *Exploiting IBM AIX Workload Partitions*, SG24-7955.

In AIX Version 7, administrators now can create WPARs that can run AIX 5.2 or AIX 5.3 in an AIX 7 operating system instance. Both are supported on the IBM POWER7® server platform and PowerHA. For details about PowerHA support, go to the following web page:

`http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/FLASH10782`

Important:

- Versioned WPARs can be non-shared system WPARs only.
- The WPAR offering is supported by IBM PowerHA SystemMirror since version 5.4.1. However, particularly in the planning phase, be careful because the combination of WPARs and PowerHA in an environment can potentially introduce new single points of failure (SPOFs).
- PowerHA does not manage or monitor the WPAR. It manages and monitors only the applications that run within the WPAR.

When deploying WPAR environments, carefully consider that you must ensure maximum availability. Potentially, the following new SPOFs can be introduced into the environment:

- The network between the WPAR host and the NFS server
- The NFS server
- The WPARs
- The operating system hosting the WPARs
- The WPAR applications

The current support of WPAR in PowerHA is oriented toward the basic WPARs:

- Currently, support is available for local (`namefs` file systems) and NFS WPARs only. WPARs can be shared or private. Versioned WPARs are also supported.
- When a WPAR-enabled resource group (RG) is brought online, all its associated resources are activated within the corresponding WPAR. The WPAR-enabled RG is associated with a WPAR based on their common name. If a resource group called `wpar_rg` is WPAR-enabled, it is associated with a WPAR with the name `wpar_rg`.
- When an RG is WPAR-enabled, all user scripts, such as application start and stop scripts must be accessible within the WPAR, at the paths that are specified in the PowerHA configuration. The user is responsible for verifying that these scripts are executable and return `0`.

- A WPAR-enabled RG can consist of some nodes that are not WPAR-capable so you do not need to upgrade all nodes of the RG to the latest AIX operating system version. And when a WPAR-enabled RG comes online on a WPAR-incapable node, it behaves as though the WPAR property for the RG is not set. However, you must ensure that all user-defined scripts are accessible at the same path as previously specified in the PowerHA configuration.
- A WPAR-enabled RG supports these resources: service label, application servers, and file systems. The service address is mandatory. The service address is allocated to the WPAR when PowerHA starts the RG.
- When a WPAR-enabled RG is deleted, the corresponding WPAR on the nodes of the RG are unaffected (that is, the corresponding WPAR is not deleted).
- All the supported resource types that are supported for a WPAR-enabled RG can be DARE-added and removed from a WPAR-enabled RG. If the WPAR property of an RG is changed through DARE (when the RG is online), the effect takes place when the RG is brought online the next time.
- PowerHA configuration verification checks that all WPAR-capable nodes of a WPAR-enabled RG have a WPAR that is configured for the RG (that is, a WPAR with the same name as the RG). If the PowerHA configuration verification is run with corrective action enabled, you are prompted to fix the WPAR-related verification errors through PowerHA corrective action. It might mean the creation of a local WPAR on all nodes that are specified in the RG modification menu.
- When a WPAR-enabled RG is brought online on a WPAR-capable node, PowerHA (which runs in the global WPAR) automatically sets up **`rsh`** access to the corresponding WPAR to manage various resources that are associated with the RG.

Important: PowerHA automatically assigns and unassigns resources to and from a WPAR as the corresponding WPAR-enabled resources come online (or go offline). You must not assign any PowerHA resources to a WPAR.

Considerations

Consider the following important information:

- PowerHA Smart Assist scripts are not supported for a WPAR-enabled RG. Therefore, any application server or application monitoring script that uses the PowerHA Smart Assist scripts cannot be configured as a part of a WPAR-enabled RG.
- Process application monitoring is not supported for WPAR-enabled RGs.
- For every WPAR-capable node that is a part of a WPAR-enabled RG and contains a WPAR for a WPAR-enabled RG, at least one of the service labels (of the WPAR-enabled RG) must be accessible from the corresponding global WPAR.

Important: Only the global instance can run PowerHA. A WPAR can be considered an RG of the type WPAR-enabled RG only.

Figure 9 on page 22 shows a highly available WPAR environment with both resilience for the NFS server and the WPAR hosting partitions. The WPAR zion is under the control of PowerHA and shares both file systems from the local host and the NFS server.

Note: The movement of WPAR RG will checkpoint all running applications that will automatically resume from the checkpoint state on the backup node (no application start up is required, but a small period of down time is experienced).

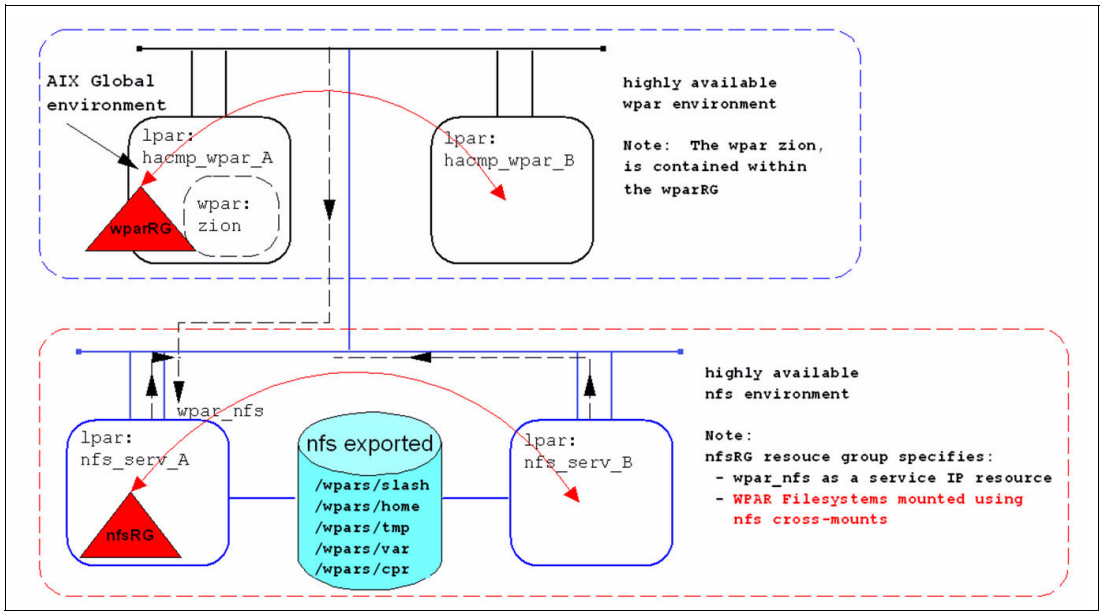

Figure 9 Highly available WPAR sample environment

Using this integration method makes WPAR support with PowerHA independent. For example, the same implementation steps can be done with any supported version of PowerHA.

```
wpar: zion

# lswpar -M zion

MountPoint        Device        Vfs     Nodename  Options
-------------------------------------------------------------
/wpars/zion       /wpars/slash  nfs     wpar_nfs  bg,intr
/wpars/zion/home  /wpars/home   nfs     wpar_nfs  bg,intr
/wpars/zion/tmp   /wpars/tmp    nfs     wpar_nfs  bg,intr
/wpars/zion/var   /wpars/var    nfs     wpar_nfs  bg,intr
/wpars/zion/cpr   /wpars/cpr    nfs     wpar_nfs  bg,intr
/wpars/zion/opt   /opt          namefs            ro
/wpars/zion/proc  /proc         namefs            rw
/wpars/zion/usr   /usr          namefs            ro

# netstat -i
Name  Mtu   Network     Address            ZoneID    Ipkts Ierrs    Opkts Oerrs  Coll
en0   1500  link#2      ea.48.f0.0.60.3              134297     0   196617     0     0
en0   1500  10.47       zion                         134297     0   196617     0     0
```

Figure 10 Example layout for WPAR: zion

More information about using WPARs with PowerHA is in the draft of *PowerHA SystemMirror for AIX Cookbook Update*, SG24-7739-01, which has an expected publish date of Q4 of 2014.

Summary

Spend considerable time in the planning stage. This is where the bulk of the documentation will be produced and will lay the foundation for a successful production environment. Start by building a detailed requirements document. Focus on ensuring the cluster does what the users want it to do and that the cluster behaves how you intend it to behave. Next, build a technical detailed design document. Details should include a thorough description of the storage, network, application, or cluster environment (hardware and software configuration) and the Cluster Behavior (RG policies, location dependencies, and more). Finally, make certain the cluster undergoes comprehensive and thorough testing before “going live” and further at regular intervals.

After the cluster is in production, all changes must be made in accordance with a documented Change Management procedure, and the specific changes must follow the Operational Procedures using (where possible) cluster-aware tools.

Following those steps from the initial start phase can greatly reduce the likelihood of problems and change after the cluster is put into production.

The following lists describe what to do, what not to do, and other information about PowerHA.

What to do:

- Must use IP address takeover (IPAT) through aliasing style networking and enhanced concurrent volume groups (VGs).
- Ensure that the hardware and software environment has a reasonable degree of currency. Take regular cluster snapshots and system backups.
- Configure application monitors to enhance availability and aid self-healing.
- Implement a test environment to ensure changes are adequately tested.
- Implement a reliable heartbeat mechanism and include at least one non IP network.
- Ensure that mechanisms are in place to send alerts via SNMP, SMS, or email when failures are encountered within the cluster.
- Implement verification and validation scripts that capture common problems (or problems that are discovered in the environment) for example, volume group settings, NFS mount and export settings, and application changes. In addition, ensure that these mechanisms are kept current.
- Make use of available PowerHA features, such as remote notification, extended cluster verification methods, “automated” cluster testing (in test only), and file collections.

What not to do:

- Do not introduce changes to one side of the cluster while not keeping the other nodes in sync. Always ensure changes are synchronized immediately. If some nodes are up and others down, ensure the change is made and synchronized from an active node.
- Do not attempt changes outside the control of PowerHA by using custom mechanisms. Where possible, use C-SPOC.
- Do not configure applications to bind in any way to node-specific attributes, such as IP addresses, host names, CPU IDs, and more. The preferred approach is to move the applications from node-to-node manually before putting them in resource groups under the control of PowerHA.

- Do not make the architecture too complex or implement a configuration that is difficult to test.
- Do not deploy basic application start and stop scripts that do not include prerequisite checking and error recovery routines. Always ensure these scripts verbosely log to stdout and stderr.
- Do not implement nested file systems that create dependencies or waits and other steps that elongate failovers.
- Do not provide root access to untrained and cluster-unaware administrators.
- Do not change the cluster failure detection rate without careful thought and consideration.
- Do not action operations such as the following example when stopping an application. This might also result in killing the PowerHA application monitor too.

  ```
  # kill `ps -ef | grep appname | awk '{print $2}'`
  ```
- Do not rely on standard AIX volume groups (VGs) if databases use raw logical volumes. Consider instead implementing big or scaleable VGs. This way, user, group, and permission information can be stored in the VGDA header and reduce the likelihood of problems during failover.
- Do not rely on any form of manual effort or intervention that might be involved in keeping the applications highly available.

Consider the following additional information:

- A written cluster requirements document allows you to carry out a coherent and focused discussion with the users about what they want done. It also allows you to refer to these requirements while you design the cluster and while you develop the cluster test plan.
- A written cluster design document describes, from a technical perspective, exactly how you intend to configure the cluster environment.
- A written test plan allows you to test the cluster against the requirements (which describes what you were supposed to build) and against the cluster design document (which describes what you intended to build). Format the test plan in a way that allows you to record the pass or failure of each test, which can help you more easily know what failed, allowing you to eventually demonstrate that the cluster actually does what the users wanted it to do and what you intended it to do.
- Do not make the mistake of assuming that you have time to write the operational documentation after the cluster is in production.
- Create a cluster HTML report by using the **`clmgr`** command.

Authors

This paper was produced by a team of specialists from around the world working at the International Technical Support Organization (ITSO), Poughkeepsie Center.

Dino Quintero is a Complex Solutions Project Leader and an IBM Senior Certified IT Specialist with the ITSO in Poughkeepsie, NY. His areas of knowledge include enterprise continuous availability, enterprise systems management, system virtualization, technical computing, and clustering solutions. He is an Open Group Distinguished IT Specialist. Dino holds a Master of Computing Information Systems degree and a Bachelor of Science degree in Computer Science from Marist College.

Alex Abderrazag is a Consulting IT Specialist with the Worldwide IBM Education Events team. Alex has over 20 years of experience working with UNIX systems and has been actively responsible for managing, teaching, and developing the Power, AIX, and Linux education curriculum. Alex is a Chartered Member of the British Computer Society and a Fellow of the Performance and Learning Institute. Alex holds a BSc (Hons) degree in Computer Science and has many AIX certifications including IBM Certified Advanced Technical Expert and IBM Certified Systems Expert HACMP (Exam Developer).

Shawn Bodily is a Senior AIX Consultant for Clear Technologies located in Dallas, Texas. He has 20 years of AIX experience and the last 17 years specializing in high availability and disaster recovery, primarily focused around PowerHA. He is a Double AIX Certified Advanced Technical Expert. He has written and presented extensively on high availability and storage. He is an IBM Redbooks® publication platinum author, co-authoring seven IBM Redbooks publications and two IBM Redpaper publications.

Daniel J. Martin-Corben is a Technical Solutions Designer for IBM UK and has been working within UNIX since he was eighteen years old. He has held various roles in the sector but has finally returned to IBM. In the early days, he worked on IBM Sequent DYNIX/ptx® as a DBA, Upon joining IBM he had his first introduction to IBM AIX and HACMP (PowerHA) and the pSeries hardware, which has dominated his career. IBM POWER8™ is his current focus, but he has extensive experience with various types of storage, which includes IBM V7000, XIV®, and SAN Volume Controller. Not only does he have strong skills and knowledge with all IBM systems, but also Solaris, Symantec, HP-UX, VMware, and Windows. He has written extensively on his IBM developerWorks® blog “Power Me Up.”

Reshma Prathap is a Certified IT Specialist in Server Systems at IBM India. She is working for the India Software Lab Operations team where she is the technical lead for virtualization of IBM System p® and System x® servers. She has over six years of experience in virtualization of System p and System x servers and four years of experience in implementing high availability solutions, especially PowerHA. She holds a Bachelor of Technology Degree in Electronics and Communication from Mahatma Gandhi University, India. Her areas of expertise include Linux, AIX, IBM POWER® Virtualization, PowerHA SystemMirror, System Management, VMware, KVM, and IBM DB2® Database administration.

Kulwinder Singh is a Certified IT Specialist at IBM GTS-TSS. He has 16 years of information technology experience. He has been with IBM since 2007. His areas of expertise include AIX, IBM System p hardware, IBM storages, IBM GPFS™, PowerHA, and IBM Tivoli® Storage Manager.

Ashraf Ali Thajudeen is an Infrastructure Architect in IBM Singapore GTS Services Delivery. He has more than eight years of experience in High Availability and Disaster Recovery Architectures in UNIX environments. As an IBM Master Certified IT Specialist in Infrastructure & Systems Management and TOGAF 9 Certified in Enterprise Architecture, he has wide experience in designing, planning, and deploying PowerHA based solutions across ASEAN strategic outsourcing accounts. His areas of expertise includes designing and implementing PowerHA and Tivoli automation solutions.

William Nespoli Zanatta is an IT Specialist from IBM Global Technology Services® Brazil. He has been with IBM for four years, supporting enterprise environments that run AIX and Linux systems on POWER and IBM System x. He has background experience with other UNIX systems and software development. His current areas of expertise include IBM PowerVM®, PowerHA, and GPFS.

Thanks to the following people for their contributions to this project:

Ella Buslovich
International Technical Support Organization, Poughkeepsie Center

Tom Weaver
IBM USA

Chris Gibson
IBM Australia

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks
- Follow us on Twitter:

  http://twitter.com/ibmredbooks
- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806
- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm
- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

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

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

This document REDP-5117-00 was created or updated on September 1, 2014.

Send us your comments in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:
  ibm.com/redbooks
- Send your comments in an email to:
  redbooks@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400 U.S.A.

Trademarks