



## Hands On with Serviceguard

H6487S K.00

# HP Training Student guide



Use of this material to deliver training without prior written permission from HP is prohibited.

Copyright 2006 Hewlett-Packard Development Company, L.P.

The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

This is an HP copyrighted work that may not be reproduced without the written permission of HP. You may not use these materials to deliver training to any person outside of your organization without the written permission of HP.

Intel386, Intel80386, Intel486, and Intel80486 are U.S. trademarks of Intel Corporation.

Intel Itanium™ Logo: Intel, Intel Inside, and Itanium registered trademarks of Intel Corporation in the U.S. and other countries and are used under license.

Java™ is a U.S. trademark of Sun Microsystems, Inc.

Microsoft® is a U.S. registered trademark of Microsoft Corp.

Oracle® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

Oracle Reports™, Oracle10™, Oracle10 Server™, are trademarks of Oracle Corporation, Redwood City, California.

Pentium® is a U.S. registered trademark of Intel Corporation.

UNIX® is a registered trademark of The Open Group.

Windows® and MS Windows® are U.S. registered trademarks of Microsoft Corp.

Printed in the USA

Hands On with Serviceguard  
Student guide  
May 2006

---

## Contents

### Overview

Student Performance Objectives.....	1
Student Profile and Prerequisites.....	6
Agenda .....	7

### Module 1 — Introduction to High Availability

1-1. SLIDE: What Causes a System to Go Down? .....	1-2
1-2. SLIDE: Causes of Failures.....	1-4
1-3. TEXT PAGE: Causes of Downtime .....	1-6
1-4. SLIDE: Not a Big Deal? You Tell Me! .....	1-8
1-5. SLIDE: Average Cost per Hour of Downtime.....	1-9
1-6. SLIDE: What Is High Availability? .....	1-10
1-7. SLIDE: Computer System Availability .....	1-11
1-8. SLIDE: Network Availability .....	1-13
1-9. SLIDE: Application Availability .....	1-15
1-10. SLIDE: Three Pillars of High Availability .....	1-17
1-11. SLIDE: 5nines Support Partnerships.....	1-18
1-12. SLIDE: High Availability Terms .....	1-19
1-13. SLIDE: High Availability Percentages .....	1-21
1-14. SLIDE: Availability Continuum Hierarchy.....	1-22
1-15. SLIDE: For More Information .....	1-23
1-16. SLIDE: Course Agenda.....	1-24
1-17. LAB: Prepare your Systems for Later Modules: Configure IP Addresses.....	1-25

### Module 2 — Introduction to Serviceguard

2-1. SLIDE: Introducing Serviceguard.....	2-2
2-2. SLIDE: High Availability with Serviceguard.....	2-3
2-3. SLIDE: Features and Benefits of Serviceguard.....	2-4
2-4. SLIDE: How Serviceguard Works.....	2-6
2-5. SLIDE: Serviceguard Packages.....	2-7
2-6. SLIDE: Redistributing Application Packages.....	2-8
2-7. SLIDE: Minimizing Planned Downtime .....	2-9
2-8. SLIDE: Serviceguard Bundle/Products .....	2-10
2-9. LAB: Installation of Serviceguard Software .....	2-11

### Module 3 — High Availability Planning

3-1. SLIDE: What Are the Risks?.....	3-2
3-2. SLIDE: Reducing the Risk ( Redundant Data ) .....	3-4
3-3. SLIDE: Reducing the Risk ( Minimizing Downtime ).....	3-5
3-4. SLIDE: Reducing the Risk ( Network Connectivity ) .....	3-6
3-5. SLIDE: Reducing the Risk ( Asymmetric Clusters and Other Issues ).....	3-7
3-6. SLIDE: Disk Configurations .....	3-9
3-7. SLIDE: LVM Mirrored JBODs.....	3-10
3-8. SLIDE: Disk Array Configuration .....	3-11
3-9. SLIDE: Highly Available SANs .....	3-12
3-10. SLIDE: Network Configurations.....	3-13
3-11. SLIDE: Redundant LANs .....	3-14
3-12. SLIDE: Normal Network Flow .....	3-16
3-13. SLIDE: Network Flow during Package Failover.....	3-17

## **Contents**

3-14. SLIDE: Routed Networks and Serviceguard Clusters .....	3-18
3-15. SLIDE: Configuration Roadmap.....	3-19
3-16. LAB: Hardware Inventory .....	3-20
3-17. LAB: Configuring and Managing LVM Boot Disks on Integrity-Based Systems (Optional).....	3-28
3-18. LAB: Configuring and Managing LVM Boot Disks on PA-Risc-Based Systems (Optional).....	3-35

## **Module 4 — LVM for Serviceguard**

4-1. SLIDE: Review of LVM Concepts.....	4-2
4-2. SLIDE: Serviceguard Configuration Using Mirrored Disks .....	4-4
4-3. SLIDE: Step 1: Configure Volume Group on First System.....	4-5
4-4. SLIDE: Step 2: Import Volume Group to Second System .....	4-7
4-5. SLIDE: VG Export and Import –s Option.....	4-9
4-6. SLIDE: LVM Issues for Using Serviceguard.....	4-10
4-7. LAB: Building a Shared Volume Group .....	4-12
4-8. LAB Solution: Building a Shared Volume Group .....	4-17

## **Module 5 — Cluster Concepts and Configuration**

5-1. SLIDE: Definition of a Cluster .....	5-2
5-2. SLIDE: Major Components of a Cluster .....	5-3
5-3. SLIDE: Network Interface Configuration .....	5-4
5-4. SLIDE: Cluster Lock Configuration Using LVM Disks .....	5-6
5-5. SLIDE: Cluster Lock Configuration Using a Quorum Server .....	5-7
5-6. SLIDE: Heartbeat Configuration .....	5-9
5-7. SLIDE: cmcl1d Process.....	5-10
5-8. SLIDE: Cluster Formation Requirements .....	5-11
5-9. SLIDE: Steps to Configure a Cluster.....	5-13
5-10. SLIDE: Step 1: Build Cluster Configuration ASCII File .....	5-14
5-11. SLIDE: Cluster Management Options.....	5-17
5-12. TEXT PAGE: The cmquerycl command.....	5-19
5-13. TEXT PAGE: Sample Cluster Configuration ASCII File Using LVM Disks for Cluster Lock Services.....	5-21
5-14. TEXT PAGE: Sample Cluster Configuration ASCII File Using Quorum Server Technology for Cluster Lock Services .....	5-25
5-15. SLIDE: Step 2: Compile and Distribute Binary File .....	5-29
5-16. TEXT PAGE: The cmcheckconf Command .....	5-30
5-17. TEXT PAGE: The cmapplyconf Command .....	5-31
5-18. SLIDE: Step 3: Start the Serviceguard Daemons.....	5-32
5-19. TEXT PAGE: The cmrunc1 Command.....	5-33
5-20. SLIDE: Cluster Configuration Procedure.....	5-34
5-21. SLIDE: Viewing the Cluster — cmviewcl Command .....	5-36
5-22. SLIDE: Checking the Cluster Log.....	5-38
5-23. LAB: Build a Cluster .....	5-39
5-24. LAB: Cluster Lock Using A Quorum Server.....	5-40
5-25. LAB: Cluster Lock Using LVM Disks .....	5-51
5-26. LAB Solution: Cluster Lock Using A Quorum Server .....	5-59
5-27. LAB Solution: Cluster Lock Using LVM Disks.....	5-71

**Module 6 — Additional Cluster Features**

6-1.	SLIDE: Additional Cluster Features .....	6-2
6-2.	SLIDE: Serviceguard Volume Groups .....	6-3
6-3.	SLIDE: Marking Volume Groups for Use in Serviceguard .....	6-5
6-4.	SLIDE: Exclusive Mode Volume Group Activation .....	6-7
6-5.	SLIDE: Cluster Formation and Reformations .....	6-8
6-6.	SLIDE: Ways to Initially Form the Cluster .....	6-9
6-7.	SLIDE: Node Failures and Node Joins .....	6-10
6-8.	SLIDE: Cluster Reformation Example .....	6-12
6-9.	SLIDE: Local LAN Card Failover .....	6-14
6-10.	SLIDE: Normal Network Flow .....	6-15
6-11.	SLIDE: Network Flow to Standby LAN Card .....	6-16
6-12.	LAB: Shared Disks/Shared Volume Groups .....	6-17
6-13.	LAB: Exiting and Joining the Cluster .....	6-21
6-14.	LAB: Local LAN Card Failover (Using Remote Equipment) .....	6-24
6-15.	LAB: Local LAN Card Failover (Using Local Equipment) .....	6-27
6-16.	LAB Solution: Shared Disks/Shared Volume Groups .....	6-30
6-17.	LAB Solution: Exiting and Joining the Cluster .....	6-34
6-18.	LAB Solution: Local LAN Card Failover (Using Remote Equipment) .....	6-37
6-19.	LAB Solution: Local LAN Card Failover (Using Local Equipment) .....	6-40

**Module 7 — Packages and Services**

7-1.	SLIDE: Packaging Concepts .....	7-2
7-2.	SLIDE: Sample Package Configuration .....	7-3
7-3.	SLIDE: Sample Configuration after Node Failure .....	7-4
7-4.	SLIDE: Package Switching .....	7-6
7-5.	SLIDE: Viewing Package Status .....	7-8
7-6.	SLIDE: Modifying Package Status .....	7-9
7-7.	SLIDE: Step 1: Build Package Configuration File .....	7-10
7-8.	TEXT PAGE: Sample Package Configuration File .....	7-14
7-9.	SLIDE: Step 2: Build Package Run/Halt Script .....	7-21
7-10.	TEXT PAGE: Sample Control Script File .....	7-24
7-11.	SLIDE: Step 3: Recompile and Distribute Binary File .....	7-34
7-12.	SLIDE: Package Scripts .....	7-35
7-13.	SLIDE: Package Owner and State .....	7-37
7-14.	SLIDE: Package Configuration Procedure .....	7-39
7-15.	SLIDE: The Package Script Log File .....	7-41
7-16.	SLIDE: Review of Commands for Controlling a Cluster .....	7-42
7-17.	REVIEW: Check Your Understanding .....	7-44
7-18.	LAB: Package Configuration #1: logger Package .....	7-47
7-19.	REVIEW Solution: Check Your Understanding .....	7-52
7-20.	LAB Solution: Package Configuration #1: logger Package .....	7-55

**Module 8 — Package Policies**

8-1.	SLIDE: Package Policies .....	8-2
8-2.	SLIDE: Failover Policies .....	8-4
8-3.	SLIDE: FAILOVER_POLICY: CONFIGURED_NODE .....	8-5
8-4.	SLIDE: FAILOVER_POLICY: MIN_PACKAGE_NODE .....	8-6
8-5.	SLIDE: Rotating Standby .....	8-7
8-6.	SLIDE: Failback Policies .....	8-8

## **Contents**

8–7.	SLIDE: Example of Automatic Failback .....	8-9
8–8.	SLIDE: Access Control Policies .....	8-10
8–9.	SLIDE: Node Fail Fast and Service Fail Fast.....	8-12
8–10.	LAB: FAILBACK POLICY .....	8-15
8–11.	LAB: Access Control Policy .....	8-17
8–12.	LAB: SERVICE_FAIL_FAST_ENABLED (Optional).....	8-19
8–13.	LAB: NODE_FAIL_FAST_ENABLED (Optional) .....	8-21
8–14.	LAB Solution: FAILBACK POLICY.....	8-24
8–15.	LAB Solution: Access Control Policy .....	8-26
8–16.	LAB Solution: SERVICE_FAIL_FAST_ENABLED (Optional) .....	8-28
8–17.	LAB Solution: NODE_FAIL_FAST_ENABLED (Optional) .....	8-30

## **Module 9—Application Monitoring Scripts**

9–1.	SLIDE: Actions Performed by Run Script.....	9-2
9–2.	SLIDE: Control Script.....	9-3
9–3.	SLIDE: Rules for Service Processes .....	9-4
9–4.	SLIDE: Application Monitoring Script.....	9-6
9–5.	SLIDE: Example: The logprog Program.....	9-8
9–6.	LAB: logprog Package Monitoring Script .....	9-9
9–7.	LAB Solution: logprog Package Monitoring Script.....	9-12

## **Module 10 — Best Practices**

10–1.	SLIDE: Best Practices.....	10-2
10–2.	SLIDE: Monitoring the syslog File.....	10-3
10–3.	SLIDE: Package Script Troubleshooting.....	10-5
10–4.	SLIDE: Package Script Log — Example 1.....	10-6
10–5.	SLIDE: Package Script Log — Example 2.....	10-8
10–6.	SLIDE: Common Cluster Configuration Issues .....	10-10
10–7.	SLIDE: Common Package Configuration Issues .....	10-12
10–8.	SLIDE: Testing Cluster Operations.....	10-14
10–9.	SLIDE: Useful Troubleshooting Commands.....	10-17
10–10.	SLIDE: The Built-In Safety Net.....	10-20
10–11.	SLIDE: Lost Cluster Lock Disk (LVM Disks) .....	10-21
10–12.	SLIDE: Monitoring Cluster and Package Resources.....	10-23
10–13.	SLIDE: Notification for Package Failure.....	10-25
10–14.	LAB: Package Monitoring with Email Notification.....	10-26
10–15.	LAB: Using the cmsetlog Command.....	10-27
10–16.	LAB: Package Monitoring with Email Notification.....	10-26
10–17.	LAB: Using the cmsetlog Command.....	10-27

## **Module 11 — Cluster and Package Online Reconfiguration**

11–1.	SLIDE: Serviceguard Online Reconfiguration .....	11-2
11–2.	SLIDE: Add a Node while a Cluster Is Running .....	11-3
11–3.	SLIDE: Remove a Node while a Cluster Is Running .....	11-5
11–4.	SLIDE: Summary of Cluster Configuration Changes.....	11-6
11–5.	SLIDE: Add a Package while a Cluster Is Running .....	11-8
11–6.	SLIDE: Remove a Package while a Cluster Is Running .....	11-9
11–7.	SLIDE: Modify a Package while the Cluster and Package Are Running .....	11-10
11–8.	SLIDE: Modify a Package while the Cluster Is Running, but the Package Is Down .....	11-11
11–9.	SLIDE: Summary: Package Modifications .....	11-12

## Contents

11–10. LAB: Online Package Addition.....	11-13
11–11. LAB: Testing Package FAILOVER_POLICY .....	11-20
11–12. LAB: Testing the Cluster-Wide ASCII File .....	11-22

### Module 12 — Highly Available NFS

12–1. SLIDE: Highly Available NFS Package .....	12-2
12–2. SLIDE: NFS Toolkit Files .....	12-4
12–3. TEXT PAGE: The nfs.flm (“File lock migration”) Script.....	12-5
12–4. SLIDE: The nfs.cntl and hanfs.sh Scripts.....	12-9
12–5. SLIDE: Using the NFS Toolkit .....	12-10
12–6. SLIDE: NFS Script Variables .....	12-12
12–7. LAB: The nfs Package .....	12-13
12–8. LAB: NFS Package Performance .....	12-21

### Module 13 — The Highly Available Oracle Database

13–1. SLIDE: Highly Available Oracle Package – Overview .....	13-2
13–2. SLIDE: Serviceguard Toolkits.....	13-4
13–3. SLIDE: Database Toolkit Contents.....	13-5
13–4. SLIDE: Example: Oracle Application.....	13-6
13–5. SLIDE: Oracle Control Script Modification.....	13-7
13–6. SLIDE: Oracle Monitor Script .....	13-8
13–7. SLIDE: Oracle Cookbook .....	13-9
13–8. LAB: Oracle Package Configuration.....	13-11
13–9. LAB: Test Oracle Package Failover.....	13-33

### Module 14 — EMS Resources and Serviceguard Packages

14–1. SLIDE: Review of Client/Server Management Architecture.....	14-2
14–2. SLIDE: MIB Structure .....	14-3
14–3. SLIDE: EMS Overview .....	14-4
14–4. TEXT PAGE: EMS Resource Class Hierarchy .....	14-5
14–5. SLIDE: Setting Up a Package to Use an EMS Resource .....	14-7
14–6. LAB: Monitoring File Systems with EMS.....	14-9
14–7. LAB: Monitoring <i>numUsers</i> with EMS .....	14-12

### Module 15 — High Availability Networking

15–1. SLIDE: Network Redundancy .....	15-2
15–2. SLIDE: Multi-Network Environment.....	15-3
15–3. SLIDE: Redundant LAN Cards .....	15-4
15–4. SLIDE: Redundant Hubs .....	15-5
15–5. SLIDE: Redundant Routers .....	15-6
15–6. SLIDE: Redundant Client Networks.....	15-7
15–7. SLIDE: Multiple IP Addresses .....	15-8
15–8. LAB: Losing Heartbeat Packets .....	15-9

### Module 16 — Rolling Upgrade Issues

16–1. SLIDE: Minimizing Planned Downtime .....	16-2
16–2. SLIDE: Rules for Rolling Upgrade .....	16-4
16–3. SLIDE: Serviceguard Rolling Upgrades .....	16-6
16–4. SLIDE: Operating System Rolling Upgrades .....	16-7
16–5. SLIDE: Cluster before Rolling Upgrade .....	16-8

## **Contents**

16–6. SLIDE: Example of a Rolling Upgrade — Running Cluster with Packages Moved.....	16-9
16–7. SLIDE: Example of Rolling Upgrade — Node1 Upgraded to HP-UX 11.23.....	16-10
16–8. SLIDE: Example of Rolling Upgrade — Install Serviceguard 11.17, Rejoin Cluster.....	16-11
16–9. SLIDE: Example of Rolling Upgrade — Run Cluster with all Packages on Node1.....	16-12
16–10. SLIDE: Example of Rolling Upgrade — Upgrade Node2 .....	16-13
16–11. SLIDE: When a Rolling Upgrade Is Not Possible .....	16-14
16–12. LAB: Performing a Rolling Upgrade .....	16-15
16–13. LAB Solution: Performing a Rolling Upgrade.....	16-19

## **Module 17 — LVM Maintenance for Packages**

17–1. SLIDE: Make a Logical Volume / File System Larger or Smaller .....	17-2
17–2. SLIDE: LVM Maintenance to a Package .....	17-3
17–3. SLIDE: Add Disk to Volume Group Owned by a Package .....	17-4
17–4. SLIDE: Add Logical Volume / File System to Volume Group Owned by a Package .....	17-6
17–5. SLIDE: Add a Volume Group to a Package .....	17-8
17–6. LAB: Increasing the Size of a Logical Volume / File System.....	17-10
17–7. LAB: Add and Remove an Unused Disk to Volume Group "vg01", Owned by the "test" Package.....	17-11
17–8. LAB: Add and Remove a Logical Volume/File System to the Volume Group Owned by the "test" Package .....	17-13
17–9. LAB: Add a New Volume Group to the "test" Package .....	17-15
17–10. LAB Solution: Increasing the Size of a Logical Volume / File System .....	17-17
17–11. LAB Solution: Add and Remove an Unused Disk to Volume Group "vg01", Owned by the "test" Package .....	17-18
17–12. LAB Solution: Add and Remove a Logical Volume/File System to the Volume Group Owned by the "test" Package .....	17-21
17–9. LAB Solution: Add a New Volume Group to the "test" Package.....	17-24

## **Module 18 — Serviceguard Manager**

18–1. SLIDE: HP Cluster Monitoring Tools.....	18-2
18–2. SLIDE: Serviceguard Manager.....	18-3
18–3. SLIDE: Welcome to Serviceguard Manager .....	18-10
18–4. SLIDE: Serviceguard Manager Login .....	18-12
18–5. SLIDE: Cluster / Node / Package Map .....	18-15
18–6. SLIDE: Initial Display .....	18-16
18–7. SLIDE: Cluster Properties.....	18-17
18–8. SLIDE: Node Properties .....	18-18
18–9. SLIDE: Node Properties .....	18-19
18–10. SLIDE: Package Properties.....	18-20
18–11. SLIDE: Package Properties.....	18-21
18–12. SLIDE: Modify Package – Nodes.....	18-23
18–13. SLIDE: Modify Package – Parameters.....	18-24
18–14. SLIDE: Modify Package – Networks.....	18-25
18–15. SLIDE: Modify Package – Services .....	18-26
18–16. SLIDE: Modify Package – EMS Resources .....	18-27
18–17. SLIDE: Modify Package – Control Script .....	18-28
18–18. SLIDE: Modify Package – Roles .....	18-29

## Contents

18-19. SLIDE: Operation Log .....	18-30
18-20. SLIDE: Online Help .....	18-31
18-21. LAB: Exploring Serviceguard Manager.....	18-32
18-22. LAB: Creating a Package with Serviceguard Manager.....	18-36

### Module 19 — Troubleshooting Scenarios and Final Review

19-1. SLIDE: Troubleshooting Scenarios .....	19-3
19-2. SLIDE: A Conflict Exists within Us .....	19-4
19-3. SLIDE: Missing the Point.....	19-5
19-4. SLIDE: Busy, Busy, Busy .....	19-7
19-5. SLIDE: The Import of This Problem Is Crucial.....	19-8
19-6. SLIDE: What Makes Johnny Run? .....	19-10
19-7. SLIDE: Fee, Fi, Foh, Fum .....	19-12
19-8. SLIDE: Care to Comment?.....	19-13
19-9. SLIDE: Thrown for a Loop?.....	19-15
19-10. SLIDE: What's in a Name? .....	19-17
19-11. SLIDE: Final Review .....	19-18
19-11. LAB Solution: Troubleshooting Scenarios .....	19-20

### Appendix A — Serviceguard Commands Summary

### Appendix B — Managing Serviceguard with HP OpenView

B-1. SLIDE: OpenView Products .....	B-2
B-2. SLIDE: SNMP Notification .....	B-3
B-3. SLIDE: SNMP .....	B-5
B-4. SLIDE: What Is Network Node Manager? .....	B-6
B-5. SLIDE: Event Configuration Integration .....	B-7
B-6. SLIDE: OpenView Operations Integration .....	B-8
B-7. SLIDE: OpenView Operations Agents.....	B-9
B-8. SLIDE: OVO Unix GUI Windows .....	B-10
B-9. SLIDE: HP OpenView Unix Java GUI with OpenView Navigator.....	B-11
B-10. SLIDE: OVO Windows GUI.....	B-12
B-11. SLIDE: OVO Template / Policy Integration .....	B-13
B-12. SLIDE: Sources of Data ... OVO .....	B-14
B-13. SLIDE: OpenView Operations and Operators.....	B-15
B-14. SLIDE: Other OpenView Products .....	B-16

### Appendix C —Serviceguard Worksheets

C-1. SLIDE: Serviceguard Worksheets.....	C-1
--	-----

### Appendix D — Integrating VxVM and Serviceguard

D-1. SLIDE: Serviceguard Disk Space Management Overview .....	D-2
D-2. SLIDE: VxVM Terminology Overview .....	D-4
D-3. SLIDE: VxVM Command Overview .....	D-6
D-4. SLIDE: Serviceguard/VxVM Software Overview .....	D-7
D-5. SLIDE: Cookbook Overview .....	D-9
D-6. SLIDE: Configure the rootdg (no vg00) .....	D-10
D-7. SLIDE: Configure the rootdg (if vg00 already exists) .....	D-12
D-8. SLIDE: Configure VxVM Disks.....	D-14
D-9. SLIDE: Configure VxVM Disk Groups .....	D-17
D-10. SLIDE: Configure a VxVM Volume .....	D-19
D-11. SLIDE: Import / Deport the Disk Group on the Other Node .....	D-22

## **Contents**

D–12. SLIDE: Manage VxVM Disk Groups.....	D-24
D–13. SLIDE: Manage VxVM Volumes .....	D-26
D–14. SLIDE: Configure a Package.....	D-28
D–15. SLIDE: For Further Study .....	D-30

## **Appendix E — Alternate Lab: New Package Configuration: `xclock` Package**

## **Overview**

The *Hands On with Serviceguard* course (H6487S) is an advanced course that teaches system administrators how to increase system availability with Serviceguard.

This course describes the concepts of high availability and fault tolerance, and discusses hardware solutions to achieve optimal availability. Other topics discussed within the framework of high availability are networking with Serviceguard, performance management, automated monitoring, backups, and monitoring Serviceguard clusters.

The course is NOT designed for software developers who will be writing applications to run in an HP Serviceguard environment. Although the course will give the software developers a good overview, they need to reference the white paper, "Designing Highly Available Cluster Applications," available through this web site:

<http://gsslweb.cup.hp.com/ServiceGuard>

## **Student Performance Objectives**

### **Module 1 — Introduction to High Availability**

- Define terms like standard availability, high availability, and fault tolerant.
- Identify single points of failure for a computer system.
- Identify causes of downtime and strategies for reducing downtime.
- Configure additional non-routable IP addresses

### **Module 2 — Introduction to Serviceguard**

- Describe three features and benefits of Serviceguard.
- Describe on a high level how Serviceguard works.
- List the components of a Serviceguard package.

### **Module 3 — High Availability Planning**

- Identify specific actions that can be taken to improve High Availability.
- Identify multiple risks (of SPOF) and list a couple of options for reducing the risks.
- Identify different disk technology for a High Availability environment.
- Identify different network configuration for a High Availability environment.
- Mirror your root volume group.

## **Overview**

### **Module 4 — LVM for Serviceguard**

- Understand how to configure a shared volume group for a Serviceguard environment.
- Understand how to export and import volume groups without having to specify disk devices.
- Understand how to activate a shared volume group in exclusive read/write access mode.
- Understand the list of operating system files that need to be modified to support shared volume groups in a Serviceguard environment.

### **Module 5 — Cluster Concepts and Configuration**

- Describe the differences between heartbeat LAN cards, stationary LAN cards, and standby LAN cards.
- Configure a Serviceguard cluster using a cluster lock disk or using Quorum Server technology.
- Configure the heartbeat-polling interval for a Serviceguard cluster.
- Describe three functions performed by the Serviceguard cluster manager daemon.
- Describe the major requirement needed to bring up a Serviceguard cluster.
- Describe Access Control Policies for a cluster
- Implement the systematic procedure for defining, configuring, and bringing up a Serviceguard cluster.

### **Module 6 — Additional Cluster Features**

- Mark a volume group as belonging to a Serviceguard cluster.
- Activate a shared volume group in an exclusive read/write access mode.
- Configure and test the local LAN failover capability of the Serviceguard cluster.
- Describe the differences between heartbeat LAN cards and standby LAN cards.
- List two methods in which a node can *join* an existing cluster.
- List two methods in which a node can *leave* an existing cluster.

**Module 7 — Packages and Services**

- Configure a basic package to run in a Serviceguard environment.
- Describe the fields in a package configuration file.
- Describe the variables in a package control script file.
- Provide the command to re-enable a package for switching.
- Interpret the output of `cmviewcl -v` as it relates to packages.
- Provide the full path name of a package-specific log file.

**Module 8 — Packages Policies**

- Use `FAILOVER_POLICY` to determine which node a package starts up on after the package fails.
- Use the `FAILBACK_POLICY` to determine which node a package starts up on after a node fails.
- Determine proper Access Control Policies for a package
- Determine when `NODE_FAIL_FAST` or `SERVICE_FAIL_FAST` should be enabled.

**Module 9 — Application Monitoring Scripts**

- Configure and implement an application monitor.
- List actions performed by the package control script.
- List three rules for Serviceguard service processes.

**Module 10 — Best Practices**

- View and interpret current activities within the cluster.
- View cluster-related activities over time.
- View activities performed by the `run` and `halt` scripts.
- Use contributed Serviceguard commands to perform troubleshooting.
- Understand the Safety Time Register.
- Replace a failed LVM lock disk.

## **Overview**

### **Module 11 — Cluster and Package Online Reconfiguration**

- Change the cluster configuration while the cluster is running.
- Add a node to the cluster while the cluster is running.
- Remove a node from the cluster while the cluster is running.
- Add a package to a running cluster.
- Delete a package from a running cluster.
- Modify specific package attributes while the package is running.
- Modify a down package while the cluster and other packages are running.
- Perform the above operations by using the HP-UX command line.

### **Module 12 — Highly Available NFS**

- Use the Serviceguard NFS toolkit.
- List files in the NFS toolkit.
- Set up a Serviceguard package using the NFS toolkit.

### **Module 13 — The Highly Available Oracle Database**

- List different Serviceguard toolkits.
- Use Oracle toolkit and cookbook.
- Configure the Oracle 10g Database to run in a Serviceguard environment.

### **Module 14 — EMS Resources and Serviceguard Packages**

- List the five EMS monitoring daemons.
- Use the `resls` command to list the available EMS resources.
- Set up a Serviceguard package to use EMS resources.

### **Module 15 — High Availability Networking**

- Connect a second network to a Serviceguard cluster with multiple levels of redundancy.
- Set up a Serviceguard node to provide LAN card redundancy.
- Configure a highly available network using redundant hubs.

- Configure a highly available network using redundant routers.
- Configure a highly available network using redundant networks.

**Module 16 — Rolling Upgrade Issues**

- Explain why rolling upgrades are carried out.
- List provisions for rolling upgrades.
- Know when rolling upgrades will fail.
- Carry out a rolling upgrade.

**Module 17 — LVM Maintenance for Packages**

- Add / remove a disk to/from a volume group owned by a package.
- Add / remove a logical volume to/from a volume group owned by a package.
- Add / remove a volume group to/from a package.
- Make a logical volume and file system (in a volume group owned by a package) larger or smaller.

**Module 18 — ServiceGuard Manager**

- Discuss the components of the Serviceguard Manager.
- Navigate the Serviceguard Manager GUI.

**Module 19 — Troubleshooting Scenarios and Final Review**

- Perform troubleshooting activities to resolve nine different Serviceguard configuration problems.
- Successfully complete 15 final review questions.

## **Overview**

### **Student Profile and Prerequisites**

This course is designed for HP-UX system administrators who develop, design, implement, and monitor Serviceguard clusters.

In order to understand the material in the Serviceguard course, to work through the labs in the course, and to work with the product in a production environment, it is essential that the student have a background in the following four areas:

1. Shell Programming — the entire process of configuring user applications to run in highly available, mission critical, environments involves writing shell scripts to start, halt and monitor the applications. A background with shell programming is essential. *POSIX Shell Programming* (H5888S), is recommended to improve in shell programming.
2. HP-UX System Administration —this course focuses on how the administration of a HP-UX server changes when it is part of a Serviceguard cluster. This course assumes the student is already familiar with standard HP-UX administration. *HP-UX System and Network Administration I* (H3064S), or *HP-UX System and Network Administration for Experienced UNIX Administrators* (H5875S), is recommended to improve in HP-UX system administration.
3. TCP/IP and basic networking — Serviceguard is designed for client/server applications and relies heavily on reassigning IP addresses from one LAN card to another, moving IP addresses from one system to another, and ARP broadcasts. *HP-UX System and Network Administration II* (H3065S), is recommended to improve in TCP/IP and basic networking.
4. LVM — The administration of Serviceguard volume groups is distinctly different from the administration of volume groups in a non-Serviceguard environment. A thorough knowledge of LVM is required to understand and confidently perform administrative tasks for volume groups in a Serviceguard clustered environment. Even as simple a task as adding a disk to a volume group in order to expand the file system/database, if not performed correctly for a clustered environment (and the standard non-clustered process is not correct), will break the cluster and leave the data vulnerable to corruption. *Hands On with Logical Volume Manager and MirrorDisk/UX* (H6285S), is highly recommended for students with limited experience with LVM. For new HP-UX administrators, H3064S and H5875S do not provide enough detail on LVM to prepare the student for Serviceguard.

## **Agenda for this 5-day class**

### **Day 1**

- Module 1      Introduction to High Availability
- Module 2      Introduction to Serviceguard
- Module 3      High Availability Planning
- Module 4      LVM for Serviceguard

### **Day 2**

- Module 5      Cluster Concepts and Configuration
- Module 6      Additional Cluster Features
- Module 7      Packages and Services
- Module 8      Package Policies

### **Day 3**

- Module 9      Application Monitoring Scripts
- Module 10     Best Practices
- Module 11     Cluster and Package Online Reconfiguration
- Module 12     Highly Available NFS

### **Day 4**

- Module 13     The Highly Available Oracle Database
- Module 14     EMS Resources and Serviceguard Packages
- Module 15     High Availability Networking
- Module 16     Rolling Upgrade Issues

### **Day 5**

- Module 17     LVM Maintenance
- Module 18     Serviceguard Manager
- Module 19     Troubleshooting Scenarios and Final Review

## **Overview**

---

# **Module 1 — Introduction to High Availability**

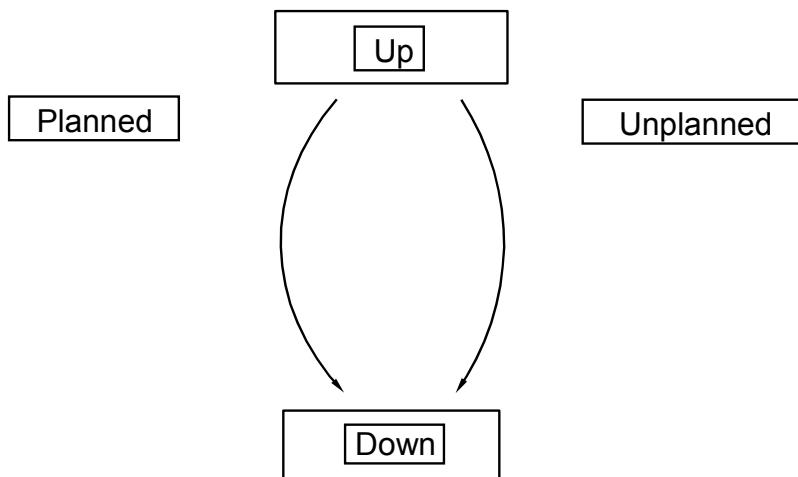
## **Objectives**

Upon completion of this module, you will be able to do the following:

- Define terms like standard availability, high availability, and fault tolerant.
- Identify single points of failure for a computer system.
- Identify causes of downtime and strategies for reducing downtime.
- Configure additional non-routable IP addresses

## 1-1. SLIDE: What Causes a System to Go Down?

### What Causes a System to Go Down?



### Student Notes

One objective of any system administrator is to keep the computer systems up and running. This objective becomes more important during business hours. It becomes even more important when the application is "mission critical" and millions of dollars are at stake when the computer system goes down.

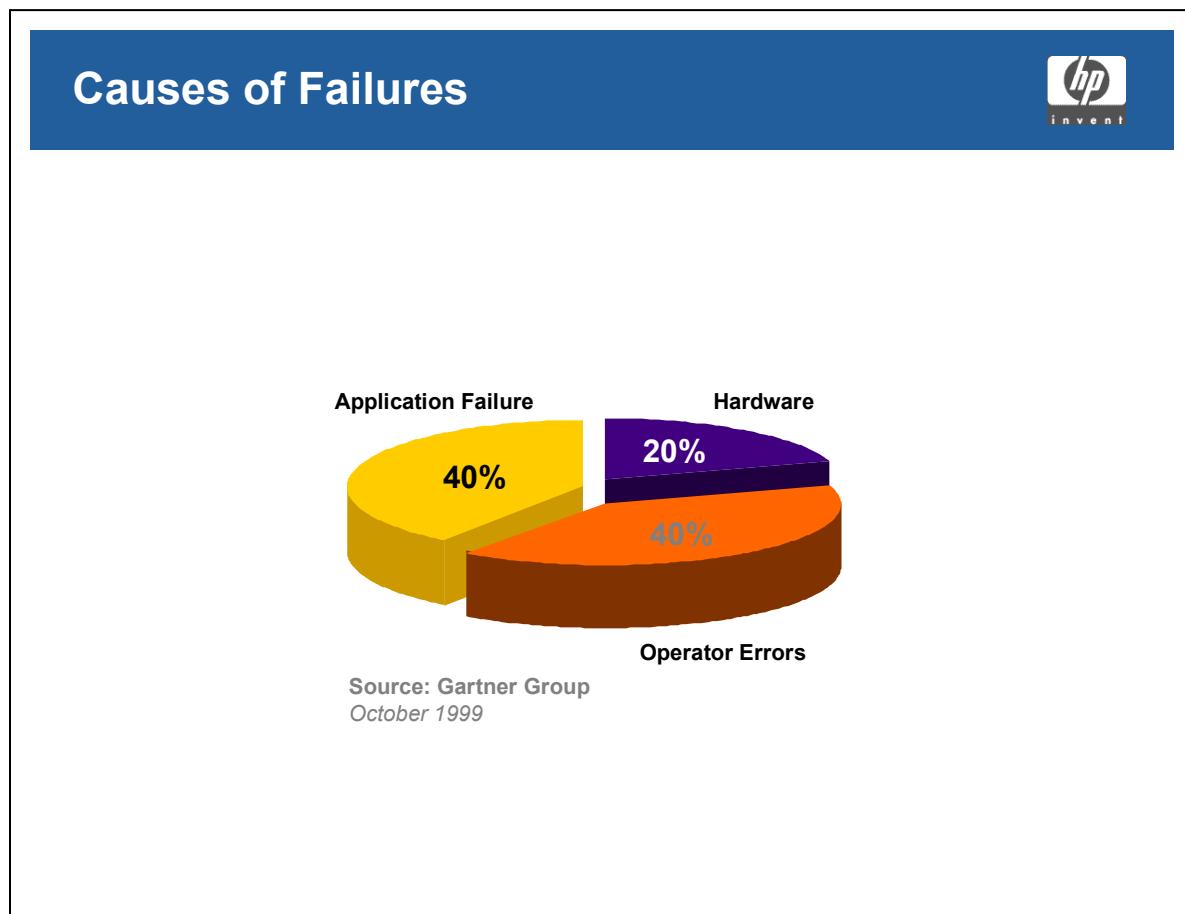
The following are some *planned* reasons why an application might become unavailable:

- To reconfigure the kernel.
- To apply patches.
- To perform software upgrades.
- To perform hardware upgrades.
- To perform disk/lvm maintenance.
- To perform full system backups.
- To perform system maintenance.
- To perform database maintenance.

The following are some *unplanned* reasons why a computer system may go down:

- Hardware failures
- System panics
- Application errors
- Power failures
- User errors
- Software defects
- Natural disasters

## 1-2. SLIDE: Causes of Failures



### Student Notes

Recent research analyzing causes of system failures show three broad categories of failures: hardware, application, and user or operator errors.

#### Operator and User Errors

The bulk of system outages today are the result of operator and user errors. Processes and procedures are just as important to high availability as the technology. The best technology will not reduce downtime if users and operators are not properly trained. Operator errors can be reduced through better tools, increased training, easier to use applications, and centralized administration.

#### Application and Software Failures

With software releases rolling out at ever increasing rates, the likelihood of software bugs also increases. Recent statistics indicate over 40 percent of system failures are software related. Though many of these bugs have patches, unless the patches are applied, the system becomes vulnerable to software failures.

## **Hardware Failures**

Hardware has traditionally been the most common cause of system failures, but recent research indicates that only 20 percent of today's system failures are hardware related. This statistic reflects the fact that hardware reliability has been increasing over the years.

Hardware failures would include such components as CPU boards, memory cards, disk controller cards, disk drives, and network controller cards.

### **1-3. TEXT PAGE: Causes of Downtime**

#### **Planned Downtime**

<b>Causes of Downtime</b>	<b>Strategies for Minimizing Downtime</b>
Backups	Hot backups and database hot backups.
Hardware/software/application reconfiguration/upgrade	Rolling upgrade with Serviceguard and self-terminating SCSI cable.
Install patch	Rolling upgrade with Serviceguard.
Database maintenance/conversion	Typically requires downtime.
Disk/LVM maintenance	On-line JFS, hot swappable disks, and RAID devices ( like HP's XP Array Series ).

#### **Unplanned Downtime**

<b>Causes of Downtime</b>	<b>Strategies for Minimizing Downtime</b>
Hardware failure, SPOF (Single Point of Failure)	SuperDome systems ( or those systems with redundant hardware components ).
CPU, memory	SuperDome systems, Serviceguard clusters, and Serviceguard clusters using Oracle 9i or Oracle 10g, with or without RAC.
Disks	LVM mirroring and disk arrays.
SCSI/Fibre Channel cards	PV links and LVM mirroring.
Network card	Backup network card through Serviceguard, and/or auto port aggregation.
Network infrastructure component	Redundant network infrastructure and dynamic routing.
Client ( PC )	Multiple PCs.
Second hardware failure ( if redundant component also fails )	Constant, on-going monitoring using scripts, SGManager, EMS, and/or OpenView products ( like Node Manager and OVO )
Power failure	UPSs and diesel generators
Software defects	Rigorous testing procedures; change control
Human error	Training, security policies, and written procedures

Natural disasters ( fire, flood, etc. )

Campus cluster, metro cluster, continental cluster

Poor performance (for some applications, poor performance is as unacceptable as downtime)

PRM, scripting to automatically move/stop applications, OVPA/OVPM, GlancePlus, and EMS

In general, the strategy for protecting against hardware failure is to build in redundant components. If the primary component fails, the redundant component automatically takes over. If the redundant component fails before the primary component is repaired, we have unplanned downtime. A failed primary component may go unnoticed without constant, ongoing monitoring.

5 Nines:5 Minutes is an HP strategy incorporating strategic alliances with Oracle, Cisco and others which for very specific environments would ensure 99.999% uptime or no more than five minutes downtime per year.

See <http://hp.com/go/ha> for more details.

- Auto port aggregation allows multiple ports ( LAN cards ) to be combined into a single link aggregate having a single IP address. This provides greater bandwidth, load balancing, and high availability.
  - Campus cluster is based on Fibre Channel hubs.
  - Continental cluster is based on database replication across a WAN.
  - Metrocluster is based on EMC SRDF, XP Continuous Access, or Continuous Access EVA.

## 1-4. SLIDE: Not a Big Deal? You Tell Me!

### Not a Big Deal? You Tell Me!





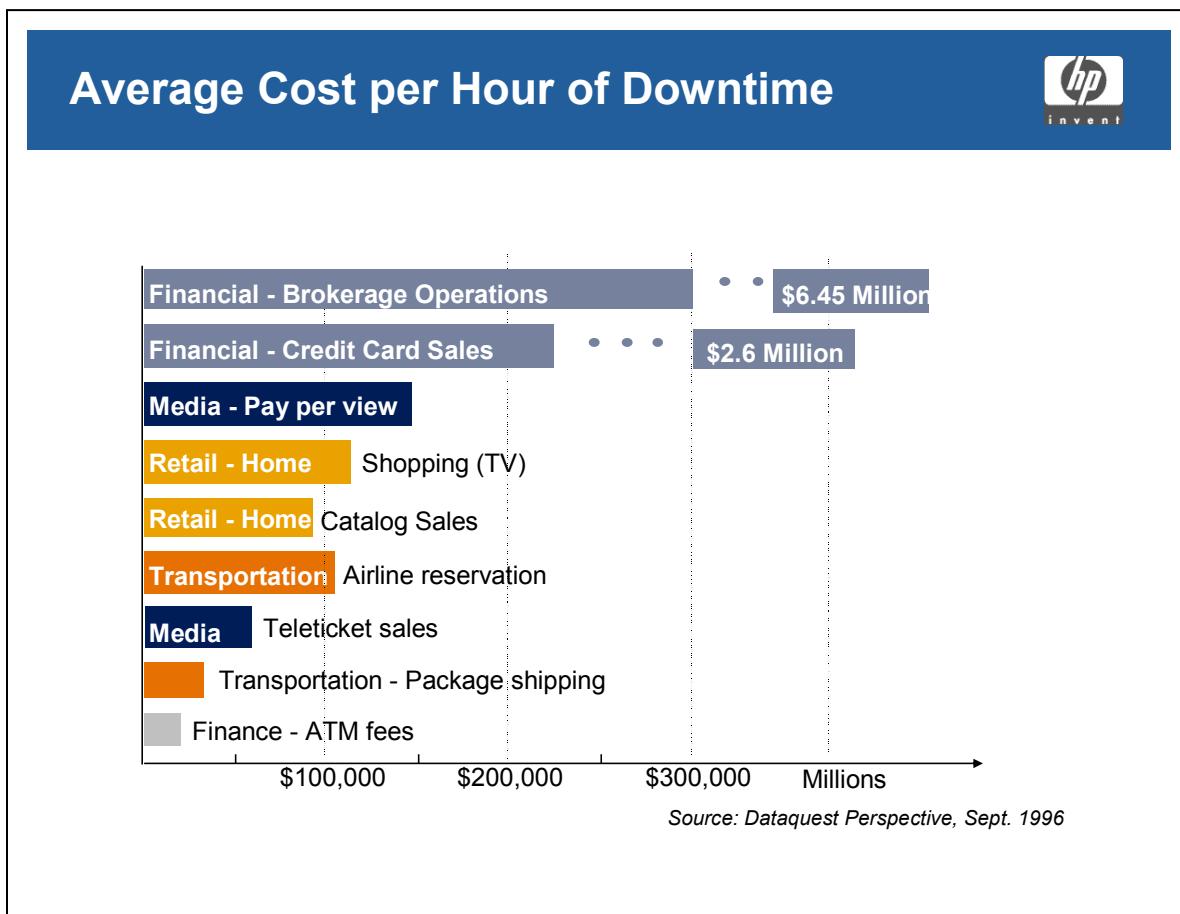
**You Tell Me!**

### Student Notes

In today's business environment, mission critical applications simply must be available all the time — or the company will end up paying the consequences in lost production, customers, and revenue.

Businesses cannot afford downtime when the competition is just a click away with e-commerce. Downtime can generate unwanted headlines and significantly affect sales revenue, customer loyalty, and stock price.

## 1-5. SLIDE: Average Cost per Hour of Downtime



### Student Notes

The cost of downtime is already high and quickly rising, making it easier than ever to justify investments in high availability ( HA ). By moving an application uptime from 99.90% uptime to 99.95% uptime, a company would eliminate nearly 4.5 hours of downtime per year. At the cost of downtime per hour as shown on the above slide, those four hours could save hundreds of thousands of dollars, depending on the company.

## 1-6. SLIDE: What Is High Availability?

### What Is High Availability?



A **system** is highly available if a single component or **resource** failure **interrupts** the system for only a **brief time**.

What is a **system**?

(Computer? Network? Application?)

What is a **resource**?

(Hardware? Software? OS? Database?)

What is a **failure**?

(Disk crash? Too many packets? Full file system?)

What is an **interruption**?

(Reboot? User reconnect? Poor performance?)

What is a **brief time**?

(Minutes? Hours? Days?)

Depends on the viewpoint . . .

**HIGH AVAILABILITY IS A DESIGN!**



### Student Notes

The above definition of high availability ( HA ) is very general and requires expansion of the terms: **system**, **resource**, **failure**, **interrupts**, and **brief time**. These definitions will vary, depending on the viewpoint. For example, system and resource tend to mean **hardware** to an administrator. To an application user, however, the **killing** of a database process is a failure resulting in loss of availability of a **software** resource.

The term **system**, in many cases, includes more than just the hardware. Depending on the "system" being kept highly available, it could include operating system resources, application processes, databases, and the ability of the users to access and connect to those resources.

Some failures can be handled transparently without any interruption ( for example, disk failure in a RAID array, single bit memory error ). Other failures can result in complete loss of service, and a restart. The priority of an HA system is to minimize the duration of the interruption ( initially minutes ) and reduce it to zero for most types of failures.

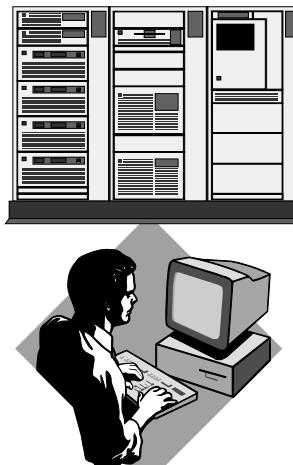
The focus in a high availability environment is *time*. It is being able to recover from any single point of failure in as brief a period as possible.

## 1-7. SLIDE: Computer System Availability

### Computer System Availability



System:	Computer
Resources:	CPU Memory Disk
Failures:	System crash Disk failure
Interruption:	System reboots Replace failed hardware
Outage Time:	Minutes to days



### Student Notes

Computer system availability is the concern of the system administrator. The primary objective in *computer system availability* is ensuring the “computer” stays up without any failed hardware resources.

In looking at the HA terms as it relates to the *computer availability*, the above slide gives examples for each term:

- The “system” being kept highly available is the computer.
- The “resources” upon which a computer depends are predominantly hardware. These resources include CPUs, memory, disk drives, controller cards, power, etc. Failure of any of these resources would cause the computer to become unavailable.
- The “failures” include events that would cause the resources to become unavailable. Examples of possible failures include CPU failures, disk drive crash, double-bit memory parity errors, power outages, etc.

## Introduction to High Availability

- The “interruptions” caused by the above failures almost always require a system reboot. In many cases, the interruption also includes the replacement of the failed hardware component.
- The “outage time” could be as short as minutes if the interruption is just a system reboot. Or it could take days if a field engineer needs to come on site to replace a failed hardware component.

## 1-8. SLIDE: Network Availability

### Network Availability



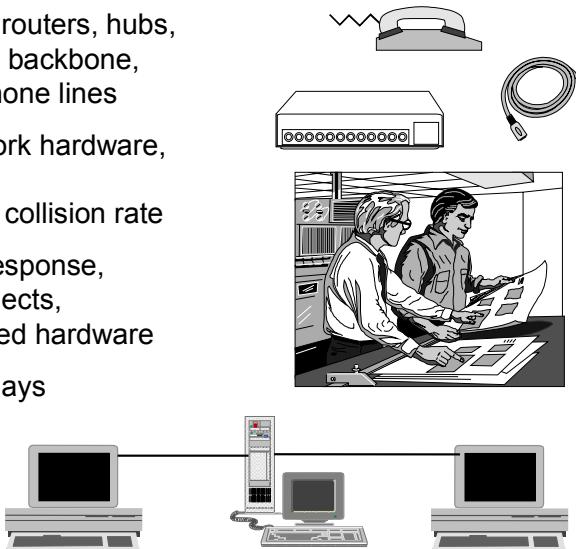
System: Network

Resources: Computers, routers, hubs, LAN cables, backbone, Modems, phone lines

Failures: Failed network hardware, Bad cables, High packet collision rate

Interruption: Slow user response, User reconnects, Replace failed hardware

Outage Time: Minutes to days



### Student Notes

Network availability is the concern of the network administrator. The primary objective in *network availability* is ensuring computers and other network devices can communicate with each other.

In looking at the HA terms as they relate to *network availability*, the slide gives examples for each term:

- The “system” being kept highly available is the network.
- The “resources” upon which the network depends include routers, bridges, hubs, network interface cards ( NICs ), etc. Failure of any of these resources would cause the network to potentially become unavailable.
- The “failures” include any resource failure. These include disconnected cables, power surges and power outages, equipment being accidentally powered off, etc.
- The “interruptions” caused by the above failures almost always results in a temporary loss of node-to-node communications.

**Introduction to High Availability**

- The “outage time” could be as short as seconds if a secondary route existed between the two nodes. Or it could take hours if an engineer needs to come on site to fix/replace a network component and only a single route between two nodes is available.

## 1-9. SLIDE: Application Availability

### Application Availability



System: Application

Resources: Computers, networks, operating system resources

Failures: System crash, Network component failure, Full file system, performance paralysis

Interruption: Slow response time, system reboots, Replace failed hardware

Outage Time: Minutes to days



### Student Notes

Application availability is the concern of the end user, the system administrator, and the network administrator. The primary objective in *application availability* is ensuring the end user has continuous access to the application ( often a client/server network-based application ).

The slide gives examples of each HA term as it relates to *application availability*.

- The “system” being kept highly available is the application.
- The “resources” upon which the application is dependent are software and hardware resources. These resource computers, networks, OS and application resources. Failure of any of these resources would cause the application to potentially become unavailable to the end user.
- The “failures” include any resource failure that would prevent the end-user from properly accessing the application. These include any previously discussed system and/or network failures, or the loss of free disk space (as in a full filesystem) or the unplanned termination of an application process.

## **Introduction to High Availability**

- The “interruptions” caused by the above failures almost always results in the application becoming unavailable to the end-user. In many cases, the interruption will require the end-user to reconnect to the application ( i.e. log back in ).
- The “outage time” could be as short as minutes or much longer, depending on the failure.

## 1-10. SLIDE: Three Pillars of High Availability

### Three Pillars of High Availability

The diagram illustrates the 'Three Pillars of High Availability'. It features a central cube with the word 'AVAILABILITY' repeated twice on its visible faces. The cube is supported by three classical-style columns, each representing a pillar. The left column is labeled 'Support Partnerships', the right column is labeled 'IT Processes and People', and the bottom column is labeled 'Technology Infrastructure'.

### Student Notes

Recognizing that only a balanced and comprehensive approach will successfully keep businesses up and running, a solid HA strategy should be built on the following three pillars, which collectively address all the major areas of downtime.

- **Technology Infrastructure.** Both hardware and software products must be available to allow the recovery of any single component failure. Often, this is the pillar which is most focused on, but it is NOT the only pillar of high availability.
- **Support Partnerships.** No one vendor will be able to provide all the HA solutions for all companies. Some vendors will specialize in hardware HA solutions, other software HA solutions, and others network HA solutions. Being able to take advantage of multiple HA solutions concurrently from multiple vendors, and having the different solutions work cooperatively together, is a key pillar of high availability.
- **IT Processes.** Even when all the technology is configured correctly and all single-points-of-failure have been properly addressed, a company may still be unable to achieve their HA goals if the solutions and HA technology is not maintained properly. Recognizing that operator and user errors are the greatest cause of outages, without good IT processes (like documented procedures and training) high availability goals cannot be achieved.

## 1-11. SLIDE: 5nines Support Partnerships

The slide has a blue header bar with the title '5nines Support Partnerships'. In the top right corner of the header bar is the HP Invent logo. Below the header bar, there are five company logos arranged in two rows: HP (top left), Cisco Systems (top right), Oracle (bottom left), BEA (bottom center), and SAP (bottom right).

**HP**  
invent

**CISCO SYSTEMS**  
EMPOWERING THE  
INTERNET GENERATION™

**ORACLE**

**bea**™

**SAP**™

### Student Notes

The 5nines:5minutes initiative encompasses several industry leaders, including HP, BEA, Cisco, EMC, Oracle, and SAP, all working together to deliver premier high availability solutions. The 5nines:5minutes initiative addresses the entire IT solution, not just individual IT components, to deliver true end-to-end availability.

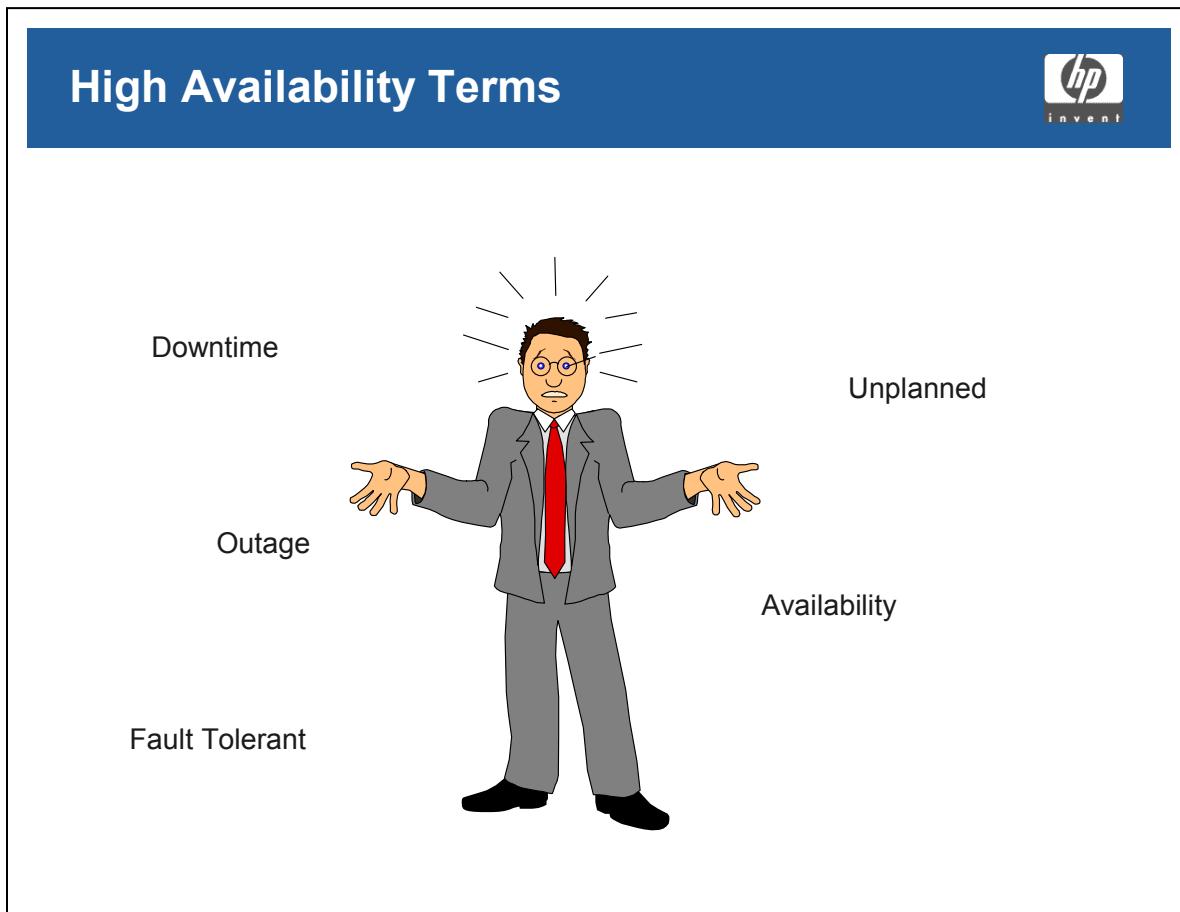
5nines:5minutes offers two solutions to address different customer needs. One, known as Mission Critical Server Suites ( MCSS ), delivers 99.95% availability, equating to only 4.5 hours of downtime per year. The other solution, for which 5nines:5minutes was named, will deliver 99.999% availability, or just 5 minutes of downtime per year.

All these solutions are for very specific applications and very specific configurations.

For more details on the 5nines:5minutes initiative, see:

<http://www.hp.com/products1/unix/highavailability>

## 1-12. SLIDE: High Availability Terms



### Student Notes

High availability terms and definitions:

- |           |  |
|-----------|--|
| 24 x 7    | An environment that must have its systems up and running 24 hours a day, 7 day a week. This environment's systems cannot be down for any reason, including maintenance or backups.   |
| 24 x 6.75 | An environment in which systems are down for 6 hours a week ( Sunday morning, from midnight to 6:00 am, for example ) to do general hardware or software maintenance.<br><br>In reality, most customers recognize there must be some downtime for these types of activities. |
| 16 x 5    | An environment in which systems are up 16 hours a day, 5 days a week ( for example, Monday through Friday, 4:00 am to 8:00 pm ). These systems may have lots of scheduled downtime. In the example cited, 8 hours a day during the week and all day on the weekends.         |

Module 1  
**Introduction to High Availability**

Downtime	Any amount of time when the application is unavailable ( planned or unplanned ). In the case of the 16 x 5 environment, there are 88 hours of downtime a week which is planned.
Planned	Downtime that is planned. This is time that the customer <i>plans</i> to bring down the system.
Unplanned	Downtime that is unplanned. This is time that the customer expects the application to be available, but it is not available due to an unplanned event or outage.
Outage	An occurrence that renders an application unavailable when it is expected to be available. An outage can occur because of hardware, software, user, or environmental problems.
Availability	The time that the application can be used during times when it is expected to be usable. Availability ignores planned or scheduled downtime and is expressed as a percentage.
Fault tolerant	A system that can tolerate any hardware failure and still keep the application available to the users. This does <i>not</i> mean that the system hardware never fails; it means the system has multiple hardware components that function concurrently, duplicating all of the computation, I/O, and so on. These systems protect against hardware failures by providing totally redundant hardware in a single system.
Highly available	A system that can recover quickly from all or most resource failures. The application may become unavailable, but only for a short period of time. Highly available systems typically contain data redundancy through mirroring or RAID, some kind of backup system hardware, a UPS for power outage protection, and multiple LANs.
Standard reliability	A system that relies only on basic hardware; there are no additional precautions taken to protect against an outage. Fault tolerant vendors claim that non-fault tolerant basic reliability results in an availability of 95 percent. HP's hardware typically provides availability in the 97–98 percent range without using any HA techniques.

## 1-13. SLIDE: High Availability Percentages

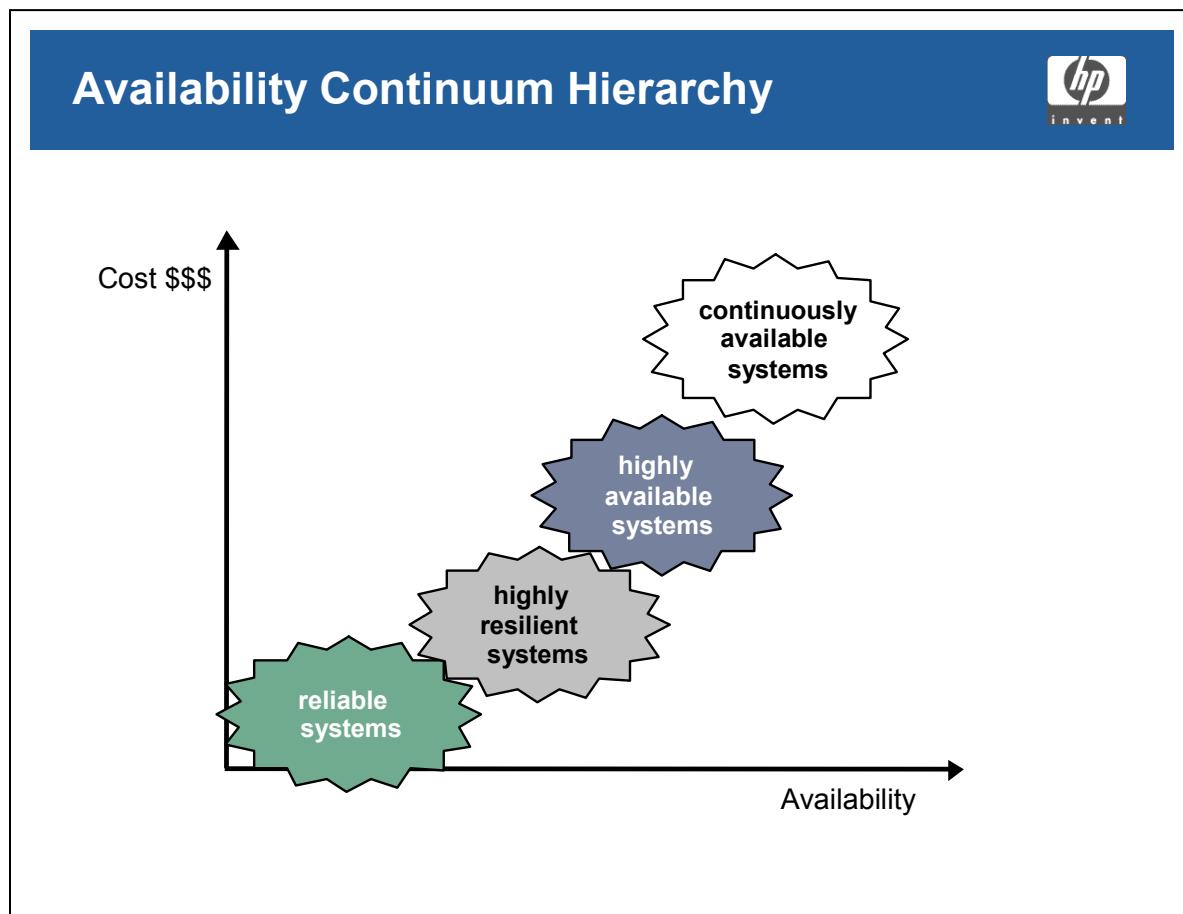
High Availability Percentages		
<u>Availability</u>	<u>Yearly Total Down Time</u>	<u>Type of System</u>
99.999	5 minutes	Fault Tolerant
99.99	50 minutes	
99.95	4.3 hours	Top High Availability
99.90	8.8 hours	
99.86	12 hours	Median High Avail
99.73	24 hours	
99.00	3.6 days	
98	7.2 days	HP Standard Avail
97	10.8 days	
96	14.4 days	
95	18 days	Most Standard Avail

### Student Notes

The slide shows the amount of downtime corresponding to the amount of availability desired. The median for highly available systems is 99.86 percent or 12 hours of downtime per year.

HP's standard reliability ( 98 percent availability ) has downtime of 7 days per year. The Serviceguard product helps to reduce downtime from *days* to *hours*.

## 1-14. SLIDE: Availability Continuum Hierarchy



### Student Notes

The above slide shows there are a number of different levels of availability. The lowest level of availability, reliable systems, implies that the reliability of the hardware determines the availability of the system.

Reliable systems typically do not recover from many single-points-of-failure (SPOF). By providing additional technology and procedures, like mirroring data and/or adding a UPS, the systems can be upgraded to "highly resilient".

A "highly resilient" system can be made "highly available" by adding the ability to recover from any SPOF, like a failed CPU or a memory failure. The Serviceguard product is designed to accomplish this objective.

Finally, "continuously available" systems allow for recovery from multiple-points-of-failure, like an earthquake destroying an entire data center. An example of a "continually available" system is HP's Continental Clusters product.

## 1-15. SLIDE: For More Information

### Additional Resources



- <http://docs.hp.com/en/ha.html>
- <http://www.hp.com/go/ha>
- <http://www.hp.com/products1/unix/highavailability>
- <http://www.itrc.hp.com>

### Student Notes

The following websites can provide further information on high availability.

<http://docs.hp.com/hpux/ha>

<http://hp.com/go/ha>

<http://www.hp.com/products1/unix/highavailability>

<http://www.itrc.hp.com>

---

## 1-16. SLIDE: Course Agenda

### Course Agenda



#### Day 1:

- Introduction to High Availability
- Introduction to Serviceguard
- High Availability Planning
- LVM for Serviceguard

#### Day 4:

- Serviceguard Manager
- Highly Available NFS
- High Availability Networking
- EMS Resources and Serviceguard Packages

#### Day 2:

- Cluster Concepts and Configuration
- Additional Cluster Features
- Packages and Services

#### Day 5:

- The Highly Available Oracle Database
- Rolling Upgrade Issues
- Troubleshooting Scenarios and Final Review

#### Day 3:

- Application Monitoring Scripts
- Package Policies
- Cluster and Package Online Reconfiguration
- LVM Maintenance for Packages
- Best Practices

## Student Notes

---

## 1-17. LAB: Prepare your Systems for Later Modules: Configure IP Addresses

### Directions

This lab will prepare us for later modules. We will configure an additional IP address for each system being used in your training class. This IP address will be used in later modules when we configure our Serviceguard cluster. Additional IP addresses will also be used in later modules when we configure Serviceguard packages.

**Your success in future modules will depend directly on your success in this lab!**

### Preliminary Steps

1. Just in case something goes wrong during this lab, make a backup copy of all of your network configuration files. There is a shell script in your `/labs` directory designed specifically for this purpose. The shell script will save a tar archive backup of your network configuration files in the file you specify. Add the `-1` option to verify your backup.

**Execute the following commands on both systems:**

```
# cd /labs
# ./netfiles.sh -s ORIGINAL
# ./netfiles.sh -l                         ← This is the letter "ell", not the number "one"
# ./netfiles.sh -l ORIGINAL                 ← This is the letter "ell", not the number "one"
```

### Part 1: Checking the Current LAN Card Configuration

1. **On both nodes**, check the current configuration of the LAN cards. How many LAN cards does your system have?

```
# lanscan
```

## Part 2: Configuring the New LAN Card Configuration

The goal of this portion of the lab exercise is to configure an additional IP address for each of the systems we will be using.

1. What IP addresses are currently assigned to your systems? Write down those values in the table below. In most cases, the `lan0` interface is unconfigured, and will remain so.

In many cases, the `lan1` interface will be configured with an IP address on the 10-net. Additional interfaces (eg. `lan2`, `lan3`, `lan4`, ...) are available for our use.

Remember that the following chart should contain only the IP addresses currently configured. There will probably be many blank spaces on your chart once you have finished this step.

```
# netstat -in
```

Just mark "none" for non-existent lan cards, and mark "nc" for lan cards not currently configured.

Lan Interface	First system	Second system
lan0		
lan1		
lan2		
lan3		
lan4		
lan5		

2. Next, add another IP address to the existing network configuration, using the next-available interface card.

Ask your instructor for the appropriate IP address to use and fill in the same chart as you used previously, but this time, also write in the new IP addresses you will be using.

Lan Interface	First system	Second system
lan0		
lan1		
lan2		
lan3		
lan4		
lan5		

3. **On both nodes**, edit the `/etc/rc.config.d/netconf` file to configure this IP address. Locate the section below and make the appropriate changes.

```
INTERFACE_NAME[0] = "lan1"
IP_ADDRESS[0] = "10.10.x.x"
SUBNET_MASK[0] = "255.255.0.0"
BROADCAST_ADDRESS[0] = ""
INTERFACE_STATE[0] = ""
DHCP_ENABLE[0] = "0"
INTERFACE_MODULES[0] = ""
```

Copy these seven lines to a spot immediately below these seven lines. After copying, change the index value from “[0]” to “[1]” for each of the seven lines. Also, make the following changes ...

- Change the `INTERFACE_NAME[1]` entry from its current value to the next available lan card interface (eg. “`lan2`”).
- Change the `IP_ADDRESS[1]` entry from its current value to the new IP address on the 10-net, as was obtained in step 2 above.
- Change the `SUBNET_MASK[1]` entry to `255.255.255.0`.
- Leave the `BROADCAST_ADDRESS[1] = ""` entry undefined.
- Leave the `INTERFACE_STATE[1] = ""` entry undefined.
- Leave the `DHCP_ENABLE[1] = "0"` entry as “`0`”.
- Leave the `INTERFACE_MODULES[1] = ""` entry undefined.

The final version of your `/etc/rc.config.d/netconf` file might then look like:

```
INTERFACE_NAME[0] = "lan1"
IP_ADDRESS[0] = "10.10.x.x"
SUBNET_MASK[0] = "255.255.0.0"
BROADCAST_ADDRESS[0] = ""
INTERFACE_STATE[0] = ""
DHCP_ENABLE[0] = "0"
INTERFACE_MODULES[0] = ""

INTERFACE_NAME[1] = "lan2"
IP_ADDRESS[1] = "10.10.y.y"
SUBNET_MASK[1] = "255.255.255.0"
BROADCAST_ADDRESS[1] = ""
INTERFACE_STATE[1] = ""
DHCP_ENABLE[1] = "0"
INTERFACE_MODULES[1] = ""
```

```
# vi /etc/rc.config.d/netconf
```

4. **On both nodes**, activate your changes.

**Special Note**

- If you are using remote equipment, then access your remote equipment via the Management Processor. Once logged into the Management Processor, connect to the Unix system or partition to which you have been assigned. From there, execute the following two steps to activate your changes.
- If you are using local equipment, then immediately execute the following two steps to activate your changes.

```
# /sbin/init.d/net stop  
# /sbin/init.d/net start
```

5. **On both nodes**, verify that your new IP address was set properly? How can you find out?

```
# netstat -in
```

6. **On both nodes**, modify your `.rhosts` files as follows:

- Do not remove any entries from your `.rhosts` file.
- To establish trust between the nodes of your cluster, add the hostname of the “other” node in your cluster.
- Ask your instructor for the hostname of the class Quorum Service machine, and add this hostname to your `.rhosts` files if it is not already there.

Add these hostnames to your `.rhosts` files.

```
# vi ~/.rhosts
```

---

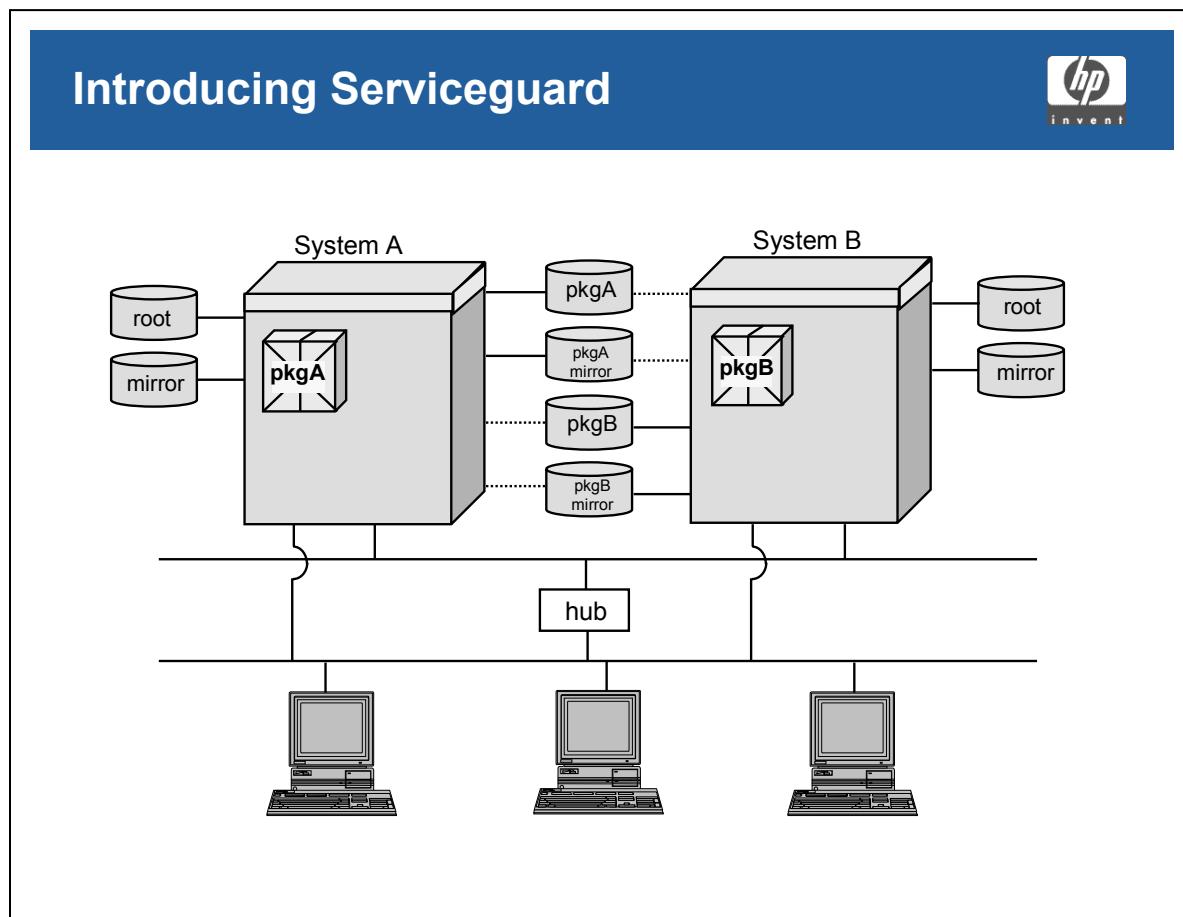
## **Module 2 — Introduction to Serviceguard**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- Describe three features and benefits of Serviceguard.
- Describe on a high level how Serviceguard works.
- List the components of a Serviceguard package.

## 2-1. SLIDE: Introducing Serviceguard



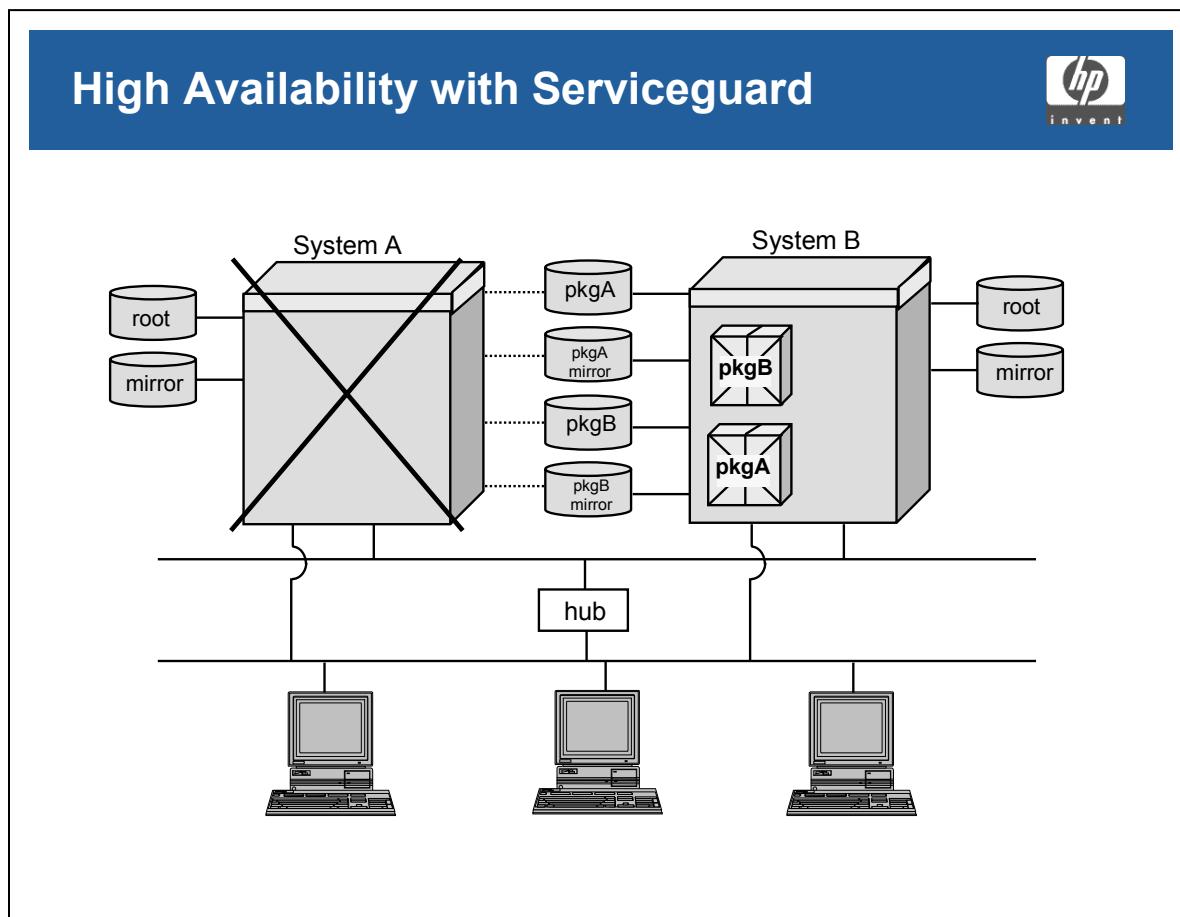
### Student Notes

Serviceguard allows you to create **high availability** clusters of HP server systems. A highly available Serviceguard environment allows applications to continue in spite of a hardware or software failure. Highly available systems protect users from software failures as well as failures from hardware failures, like failures of CPU, disk, or network components. In the event that one component fails, the redundant component takes over.

A Serviceguard cluster is a networked grouping of HP business servers having sufficient redundancy of software and hardware that a single-point-of-failure will not significantly disrupt service. Applications resources are grouped together in **packages**, and in the event of a failure of an application/package resource, Serviceguard can automatically transfer control of the application to another node within the cluster.

In the above slide, SystemA is running PackageA and SystemB is running PackageB. Each package has a separate group of disks associated with it, containing data needed by the package's applications, and a mirror copy of the data. Note that both nodes are physically connected to both groups of mirrored disks. In the slide, SystemA is shown with exclusive write access to the top disks (solid lines) and System B is shown as connected without access to the top disks (dotted lines). Similarly, SystemB has exclusively access to the bottom disks and SystemA is connected, but does not have access to the bottom disks.

## 2-2. SLIDE: High Availability with Serviceguard



### Student Notes

Under normal conditions, Serviceguard simply monitors the health of the cluster and package components. When Serviceguard detects a node or package failure, it will transfer control of the package to the next available node in the cluster. This is the situation shown on the above slide.

Once a package is transferred to a second node, it remains there as long as the second node continues running ( by default ). However, the package can be configured to automatically return to its primary node as soon as the primary node comes back online. Or, the system administrator can manually transfer the package back to the primary node at any appropriate time.

Serviceguard is designed to work with other high availability (HA) products to provide even higher levels of availability. Some of the other HA products include:

- **MirrorDisk/UX:** eliminates SPOF of the disk subsystem
- **Event Monitoring Service (EMS):** allows monitor and detection of non-Serviceguard resources
- **Disk Arrays:** provides RAID level data protection
- **UPSS:** eliminates failures related to power outages

## 2-3. SLIDE: Features and Benefits of Serviceguard

### Features and Benefits of Serviceguard



#### Features

- Highly Available Cluster ( applications recover to alternate nodes in < 60 seconds -- often in < 10 seconds )
- LAN failure protection ( fast local switch to standby LAN adapter inside same node )
- Application Packages allow all resources for a package to be defined in one place
- Automatic cluster reconfiguration after a node failure
- Intelligent cluster reconfiguration after a node failure
- No idle resources
- Facilitates online hardware and software updates
- Any mix of PA-RISC and / or Integrity Servers

#### Benefits

- Applications remain available to users, even after a hardware or software failure
- LAN card failures do not cause an application outage
- Applications can be moved easily and transparently without client reconfiguration
- No manual user intervention is needed to recover from a node failure
- Data integrity is preserved during a node failure ( i.e. no "split-brain" syndromes )
- Every node runs a production application
- Applications available during hardware and software upgrades
- Flexible load balancing during failover

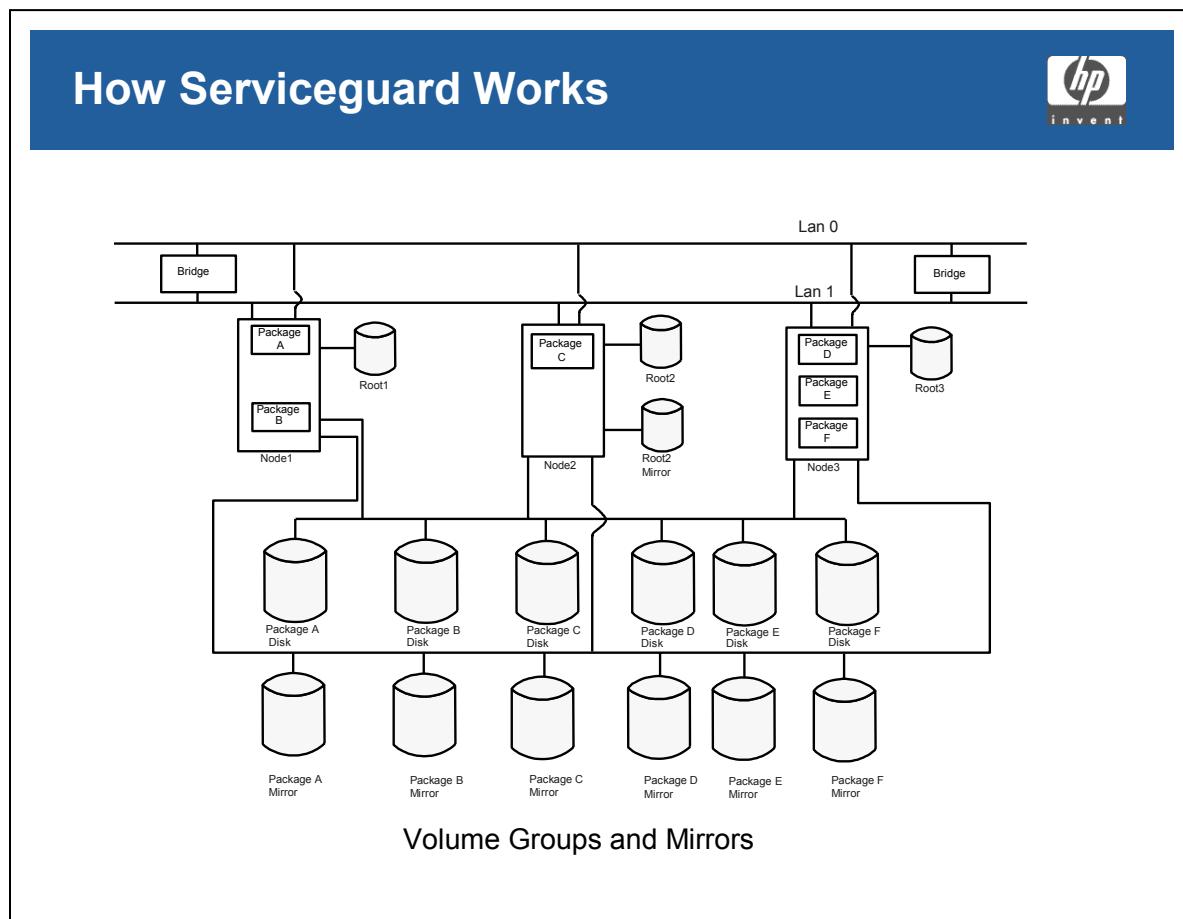
### Student Notes

Features and benefits of Serviceguard include the following:

- *High cluster availability:* Potentially, any node in the cluster can run the shared applications. If a node goes down, all shared applications on the node will move to the other available nodes within the cluster.
- *LAN failure protection:* Every LAN interface card (LANIC) is capable of having a corresponding standby LANIC. If a primary LANIC fails, all IP addresses configured on that card would move to the standby card.
- *Application transparency:* Clients (workstations) will not be able to tell on which node an application is running.
- *Fast application switching:* The amount of time between when an application initially fails and when the application is restarted on another node is under 60 seconds. The Journal File System (JFS) is highly recommended in order to optimize the failover time by reducing the amount of time needed to repair the file system.

- *Easy and flexible application management:* Applications can be easily stopped and restarted on nodes within the cluster. Tools to monitor the state of the cluster and applications within the cluster exist in both a GUI and character-based mode.
- *Intelligent reconfiguration:* In the event of a failure, the cluster will automatically reconfigure itself to make the application available. No user interaction is required for reconfiguration.
- *Flexible load balancing:* Upon node failure, the workload on the failed node can be distributed across surviving nodes in the cluster. For example, if a node had three applications and the node failed, the applications could be spread across three different systems within the cluster.
- *No idle resources:* Every system is a production system, every system within the cluster can be running mission critical applications. No system sits idly, waiting for another system to fail.
- *Online software update:* Systems within the cluster can now have their software updated without having their applications unavailable to the users. The applications could run on alternate nodes while the system is being software upgraded.
- *Any mix of PA-RISC and / or Integrity Servers:* A cluster can have a mix of business server types, both PA-RISC and Integrity. For example, one or more partitions from a large Superdome system could be in the same cluster as an rp8400.

## 2-4. SLIDE: How Serviceguard Works



### Student Notes

A **Serviceguard cluster** is a networked group of nodes (hosts), which monitor each other, in order to ensure that interruptions to the availability of selected services running on these nodes are kept small.

The state of any node or network interface within a cluster can be monitored from any node within the cluster. For example, if `/dev/lan0` has failed on node 1, that information will be obtained by each node within the cluster. The Serviceguard daemon monitors the health of the cluster, all nodes in the cluster, and all network resources.

A cluster may contain up to 16 nodes. The original node is the first node on which a package is running before Serviceguard initiates a fail-over. The adoptive node is a node to which Serviceguard could transfer control of a package. A package can have several adoptive nodes. The node on which a package is currently running is said to have package custody.

## 2-5. SLIDE: Serviceguard Packages

### Serviceguard Packages

The diagram shows a central gray rectangular box containing three smaller gray rectangular boxes labeled "pkgA", "pkgB", and "pkgC". Below this central box is a white rectangular box with a black border containing the following text:  
Service Processes:  
App\_Process\_1  
App\_Process\_2  
Middleware\_1  
Middleware\_2  
  
Volume Groups:  
/dev/vg01  
/dev/vg02  
  
IP Address:  
156.152.194.134

### Student Notes

A Serviceguard package defines an application along with its programs and resources. It is dependent on such resources as disks, network addresses, files, and services that may be related to an application. All of these resources would move together to another system in the event of a failure on the original system or network. The package configuration file lists all the nodes that the package is able to run on within the cluster. Packages provide the mechanism for switching a highly available service (and all it is dependent on) as a unit between nodes in response to failures. If a node has multiple packages running on it, the separate packages can be switched to separate nodes, spreading the load of the original node.

No more than one package can depend on a volume group. You can specify if you want control of a package to be transferred or if you want the package to be halted in the event of a failure. The adoptive SPU must have access to all files and services needed to run the application.

Packages, therefore,

- may use disk resources,
- may use network resources,
- must include at least one process, and commonly include many processes.

## 2-6. SLIDE: Redistributing Application Packages

### Redistributing Application Packages

The diagram illustrates the process of redistributing application packages in a cluster after a node failure. It shows four nodes represented as boxes, each containing several application packages (represented as smaller boxes). Node 1 (bottom left) contains packages pkgA, pkgB, and pkgC. Node 2 (top left) contains packages pkgE, pkgF, and pkgA. Node 3 (top right) contains packages pkgB, pkgG, and pkgH. Node 4 (bottom right) contains packages pkgI, pkgJ, and pkgC. A curved arrow points from Node 1 to Node 2, indicating that package pkgA is being moved from Node 1 to Node 2. Another curved arrow points from Node 1 to Node 4, indicating that packages pkgB and pkgC are being moved to Node 4. A callout box labeled "If Node 1 fails..." is positioned below Node 1, pointing towards Node 4.

### Student Notes

In the event of a node failure, an application package will be adopted by another node in the cluster. Later, the failed node is able to rejoin the cluster. Application packages can remain running on the adoptive node or can be moved back (by the Serviceguard administrator using either the command line or Serviceguard Manager) to the original node after the original node is back on line.

## 2-7. SLIDE: Minimizing Planned Downtime

### Minimizing Planned Downtime

The diagram illustrates a cluster of nodes being upgraded from HP-UX 11.11 to HP-UX 11.23. It shows two main stages of the upgrade process:

- Initial State:** A cluster of nodes labeled "HP-UX 11.11". Inside this cluster, there are two groups of nodes:
  - A group containing nodes A, B, C, and D.
  - A group containing nodes E, F, G, H, I, and J.
- Upgrade Stage 1:** An external box labeled "Upgrade from HP-UX 11.11 to HP-UX 11.23" is connected to the cluster. It is shown upgrading the group of nodes A, B, C, and D to "HP-UX 11.23".
- Final State:** The cluster has been fully upgraded. The original group of nodes A, B, C, and D is now labeled "HP-UX 11.23". The second group of nodes E, F, G, H, I, and J remains labeled "HP-UX 11.11".

### Student Notes

We will spend minimal time in this discussion right now. In a later module, we will discuss this “rolling upgrade” technique in detail.

For now, suffice it to say that we do the following to minimize planned downtime with Serviceguard:

1. Move packages to alternate nodes.
2. Perform hardware or software upgrade/maintenance.
3. Move packages back and roll through cluster.

## 2-8. SLIDE: Serviceguard Bundle/Products

### Serviceguard Bundle/Products

The diagram illustrates a Serviceguard cluster setup. Two server nodes are connected to a group of four database cylinders. A central box labeled "Serviceguard Bundle - B3935DA" contains two products: "Cluster Monitor Product" and "Package Manager Product". To the right of this bundle are six pink squares representing additional products: "EMS Product", "SG Toolkits", "SG Patches", "SGMgr Product", and two other unlabeled pink squares.

### Student Notes

The Serviceguard product is located on the HP-UX 11.00 and 11i Application media.

The core Serviceguard products are bundled together and are automatically installed when the administrator installs the HP-UX 11i Mission Critical Operating Environment. The core products are Cluster Monitor and Package Manager. Additional software can supplement these products, including the EMS product, various Toolkits, and related Serviceguard patches.

In addition, a standard Serviceguard installation should also include:

- OnLineJFS
- LVM MirrorDisk/UX
- PRM (Product Resource Manager) for managing system performance
- EMS (Event Monitoring Service)
- Serviceguard Manager

All these products (Serviceguard and all its supplemental products) must be installed on each system in the cluster.

## 2-9. LAB: Installation of Serviceguard Software

### Directions

#### Special Note

When these systems were loaded for our Serviceguard class, HP-UX 11i including the "Mission Critical Operating Environment" was installed. This Operating Environment includes the "OnLine JFS" software, the "MirrorDisk/UX" software. Of course, it also includes the Serviceguard software itself. This Operating Environment also includes most (but not all) of the additional software needed for this class. The following lab installs the remaining Serviceguard-related software. This software needs to be installed on all systems which later will form a Serviceguard cluster.

1. List all existing software bundles on the system. Verify that Serviceguard is installed.

**Answer:**

```
# swlist -l product | grep -i guard
```

2. Run the Software Distributor `swinstall` command.

**Answer:**

```
# swinstall
```

3. At the `Specify Source` window, accept the default host name and default depot name, which is `/var/spool/sw`.
4. At the `Software Selection` window, select the following products:
  - B8325BA Serviceguard Manager (version A.05.00)

**Module 2**  
**Introduction to Serviceguard**

5. Once the software has been selected, go to the **Action** menu and select **Install**.
6. When the analysis phase completes, the status of the installation will change to **READY**. Select **OK** to continue with the installation.
7. When the installation phase completes, the status of the installation will change to **COMPLETE**. Select **Done** to exit the window.
8. Select **EXIT** from the **File** menu to exit Software Distributor.
9. Verify that the above software product has indeed been properly installed.

**Answer:**

```
# swlist -l product | grep -i guard           ← For Serviceguard Manager
```

---

## **Module 3 — High Availability Planning**

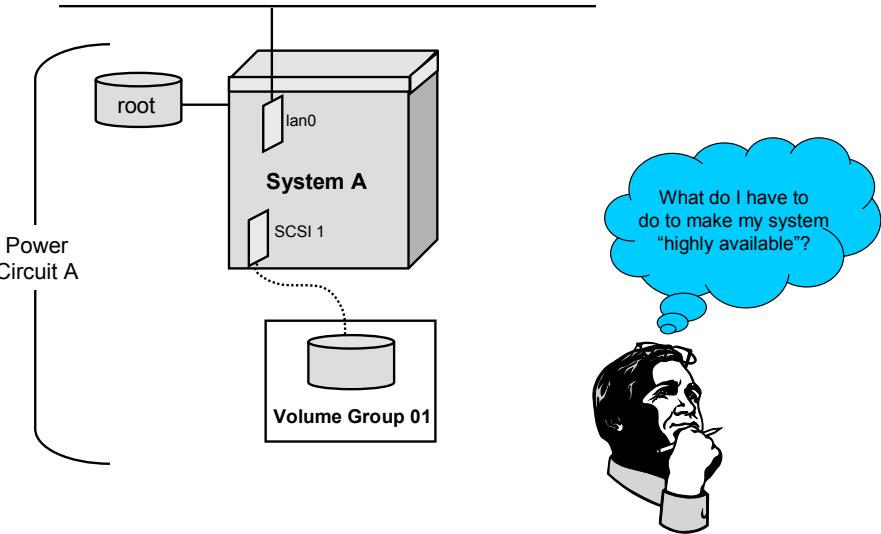
### **Objectives**

Upon completion of this module, you will be able to do the following:

- Identify specific actions that can be taken to improve high availability (HA).
- Identify multiple risks (SPOF's) and list options for reducing the risks.
- Identify different disk technology for a high availability environment.
- Identify different network configuration for a high availability environment.
- Mirror your root volume group.

### 3-1. SLIDE: What Are the Risks?

## What Are the Risks?



What do I have to do to make my system "highly available"?

### Student Notes

The simple drawing on the above slide illustrates several common Single-Points-Of-Failure (SPOFs). A common perception of SPOFs is that they are only hardware related. Items often listed as SPOF include:

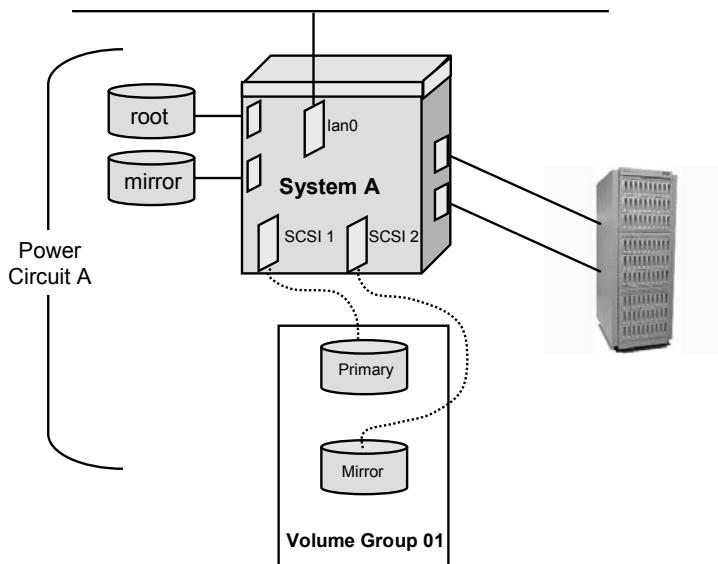
- Disk drives
- Controller cards
- LAN cards and cables
- CPU
- Power supply

But, SPOFs can also include bugs in the operating system, poorly set kernel parameters, and/or unknowledgeable system administration personnel. Other aspects to consider are not only the availability of the system or application(s) but response times and performance. The performance of a server may be poor because of heavy demand or bad configuration. From the viewpoint of end-users and client applications, it might as well be unavailable.

HA technologies and plans that effectively use these technologies can prevent these situations. HA design should consider the possible management and control potential, as well as the *reactive to failure* elements.

### 3-2. SLIDE: Reducing the Risk (Redundant Data)

## Reducing the Risk ( Redundant Data )



### Student Notes

The first HA consideration is usually related to *data* and disk drives. A design incorporating some form of hardware RAID technology or software mirroring is common. We will compare the features and benefits of these two approaches later.

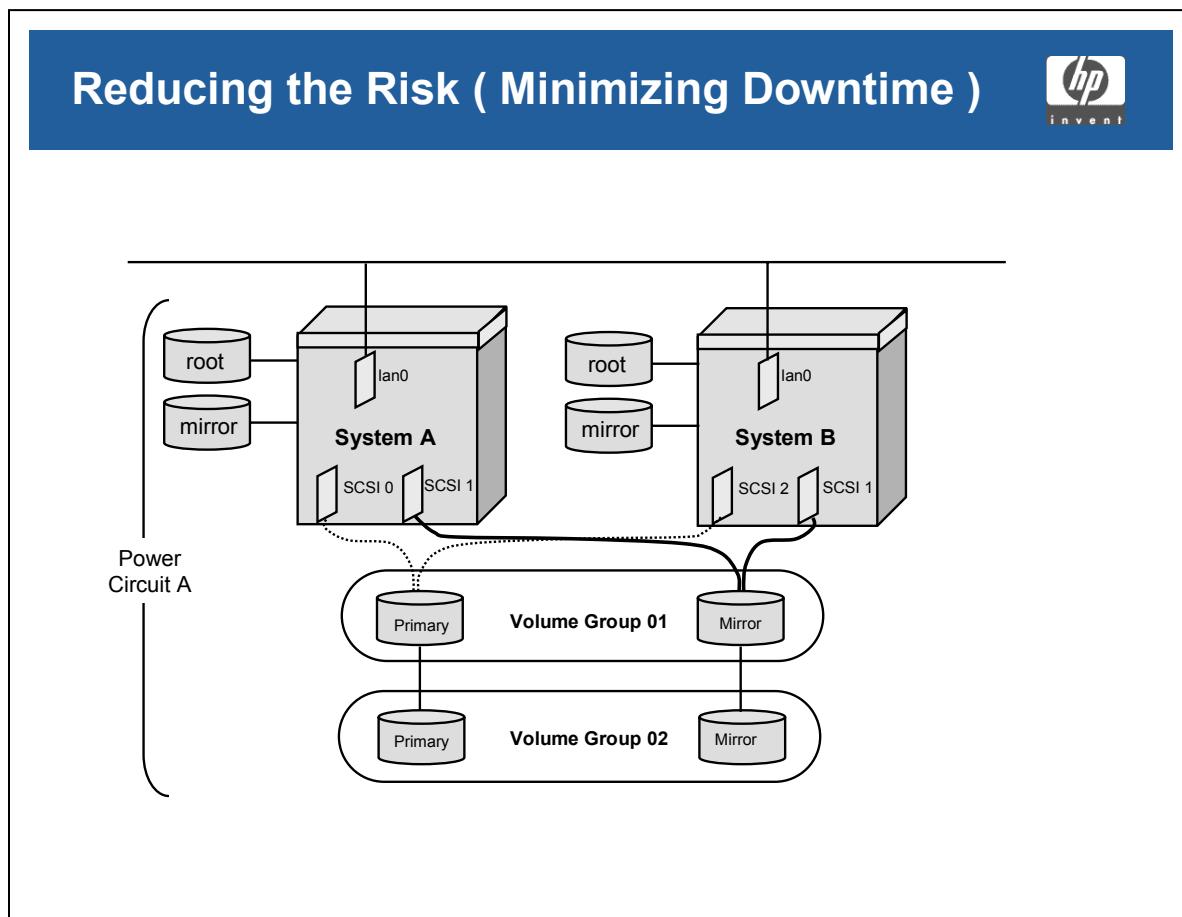
From both an HA and performance perspective, the design should include multiple controller paths and cables. LVM PV links are only used in the event of a primary path timeout, and do not load share. Good RAID array design and configuration will utilize all controllers, each acting as PV link paths for the other, resulting in HA and load sharing.

What type of RAID array and level to use cannot be taken in isolation. We must consider other technologies. (Serviceguard and Serviceguard SGeRAC edition, for example, can require specific *lock disk* types under certain circumstances.)

Other pertinent questions which should be asked relative to protecting data:

- Is online backup a requirement?
- How easy is it to replace a failed disk drive?
- Is it necessary to mix RAID levels, and tune for performance?

### 3-3. SLIDE: Reducing the Risk (Minimizing Downtime)



#### Student Notes

If there is only one computer, then there is a SPOF. Utilizing Serviceguard (or, in some circumstances, Serviceguard SGeRAC), an HA cluster can be designed to allow other systems to provide a support for the applications and services. These additional nodes will require access to the disk drives and networks. The HA software *monitors* the health of the nodes and services, and takes appropriate action on failures.

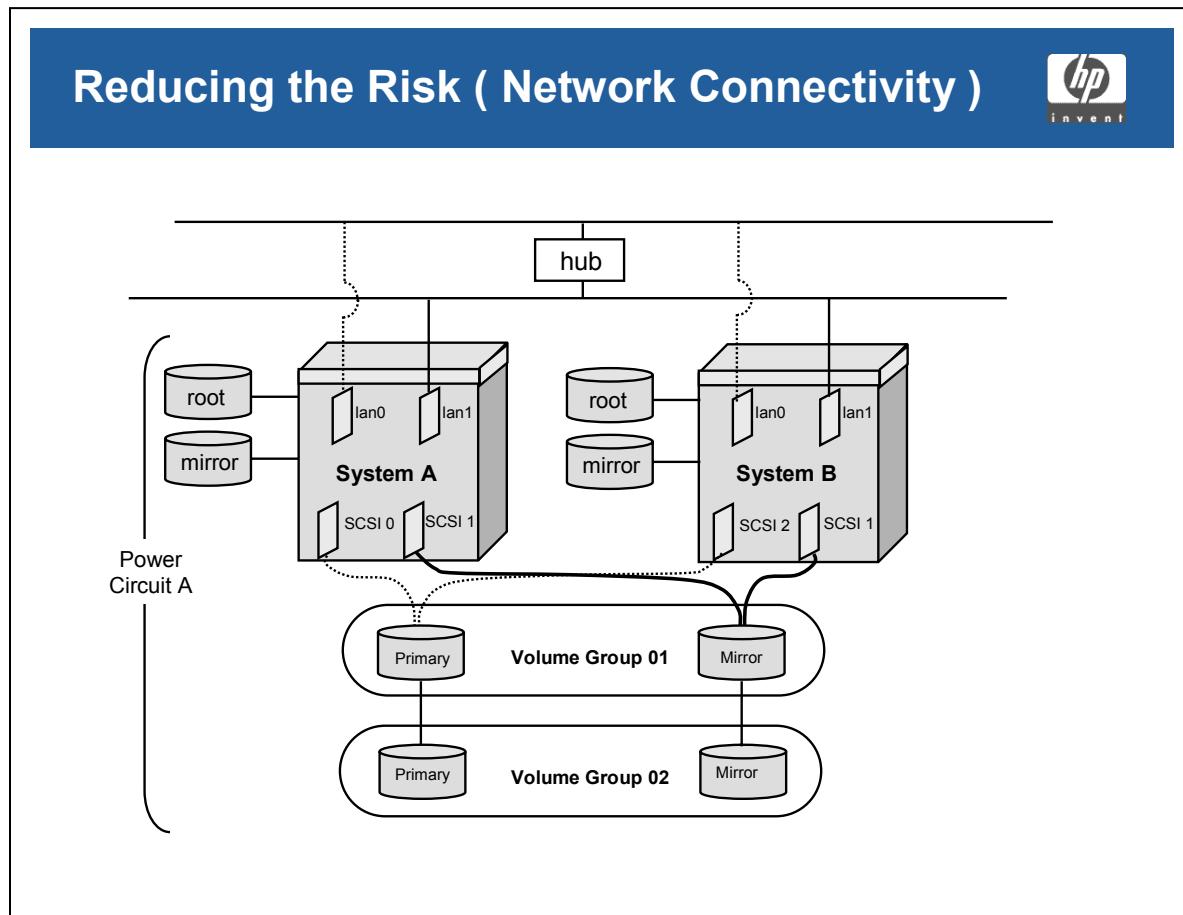
These failures can be:

- Hardware – e.g. CPU failure.
- Software – e.g. termination of a process.
- Monitored resource event – e.g. task wait time exceeds a defined value.

This design also facilitates the easy *movement* of applications between systems, allowing system upgrades (both hardware and OS software) to be performed with minimum planned downtime. (The downtime would just be the time taken to close the application on one system and restart it on the other.)

Depending on the shared disk technology, it is also possible to perform backups from the second system.

### 3-4. SLIDE: Reducing the Risk (Network Connectivity)



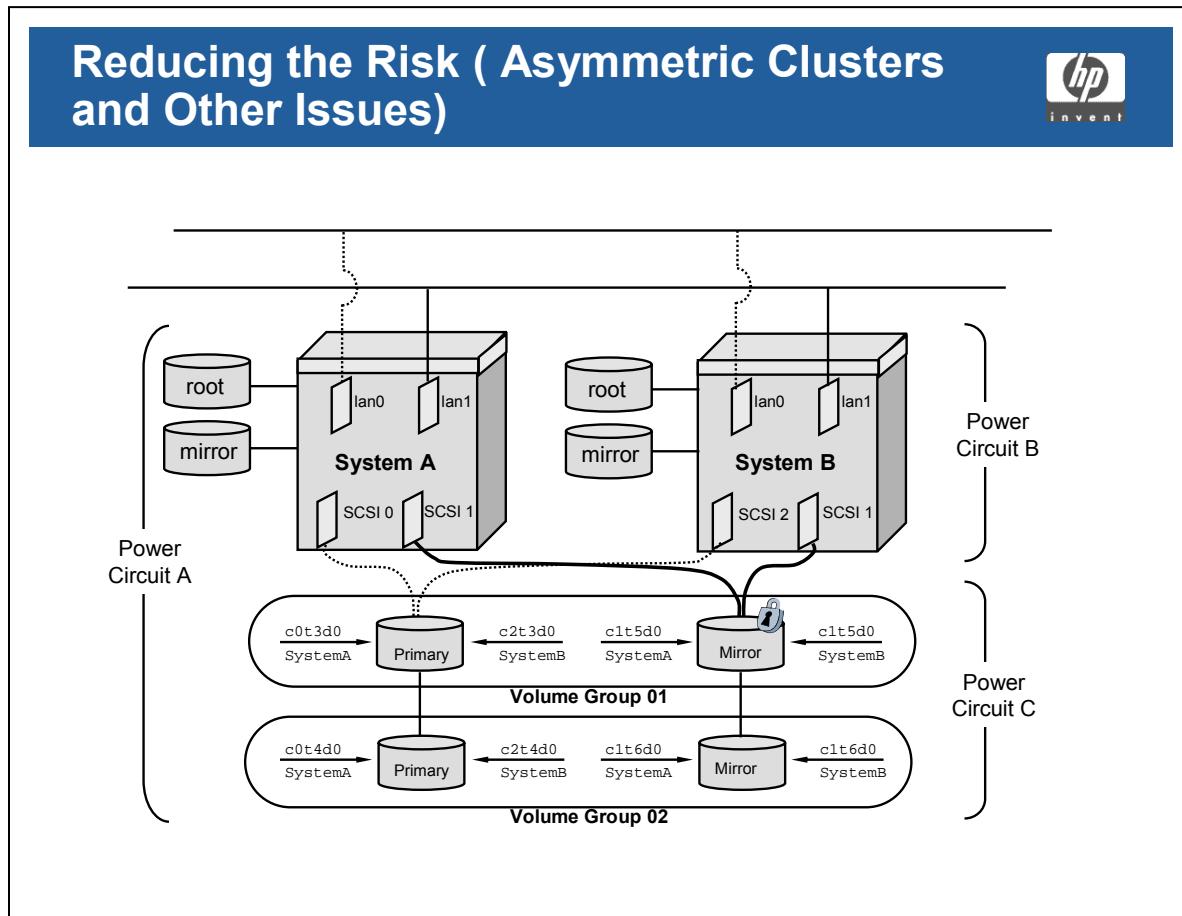
### Student Notes

In an HA cluster, clients will access the services via the network. This is another SPOF that can be accommodated by the cluster software. Using multiple LAN controller cards, or LANICs, bridges, switches, and cabling, etc., a system can switch rapidly to a *standby* LANIC without any perceived interruption by clients.

Applications and services can also be *linked* to network availability such that they will relocate to another system if such a standby is not available and the *monitored* network fails on this system.

Always consider the availability from the client through to the *service*, including redundant *routes*.

### 3-5. SLIDE: Reducing the Risk (Asymmetric Clusters and Other Issues)



#### Student Notes

In addition to reducing physical risks, a solid high availability plan must also address:

- Potential power outages.
- HA training for operators and support personnel.
- Agreed upon procedures for recovering from various failures.
- The availability of documentation for the HA products.
- A well configured cluster which follows the "best practices" for cluster configuration.

The above slide provides an example of what can happen when a cluster is not configured using "best practices." One of these "best practices" is to make the cluster symmetrical. One part of symmetry is that device file names used to reference the shared disks should be the same across all nodes in the cluster.

Module 3  
**High Availability Planning**

Because the two systems use two different instances of SCSI controllers to share a SCSI bus, the cluster is asymmetric. The device file names are different for these shared disks, where the device file name is dependent on the system from which the disk is accessed. For example, If a disk is accessed from SystemA, the device file name is `c0t3d0`, but if it is accessed from SystemB, the same disk has a device file name of `c2t3d0`.

## 3-6. SLIDE: Disk Configurations

### Disk Configurations



#### Possible Disk Configurations for a Serviceguard Cluster Include:

- LVM Mirrored JBODS
- Disk Array Configuration
- Storage Area Networks

### Student Notes

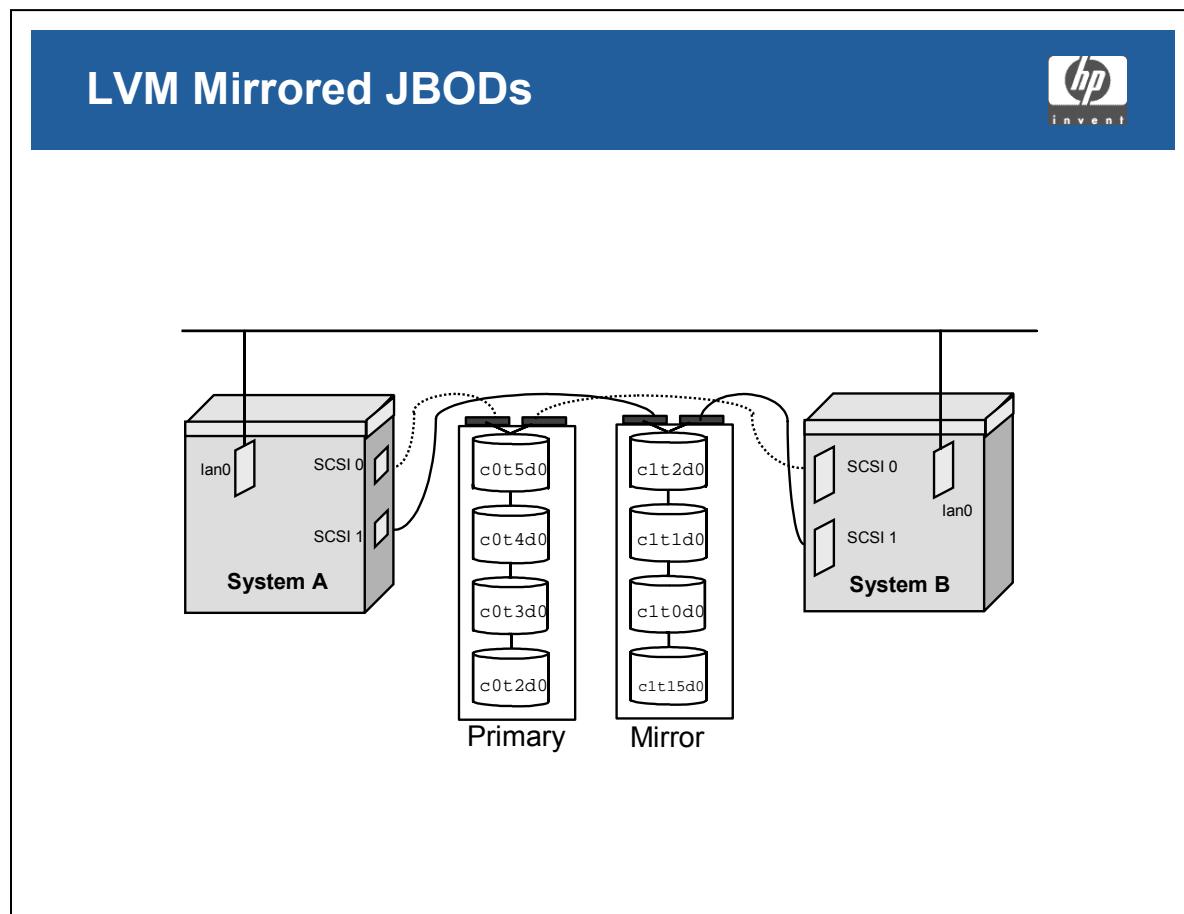
The above slide outlines three different disk technologies/configurations supported by Hewlett Packard in a Serviceguard environment.

Each disk technology has its own set of advantages and disadvantages. These include:

- Performance
- Price
- Capacity
- Ease-of-use

The advantages and disadvantages of each of these disk technologies are discussed in the following pages.

### 3-7. SLIDE: LVM Mirrored JBODs

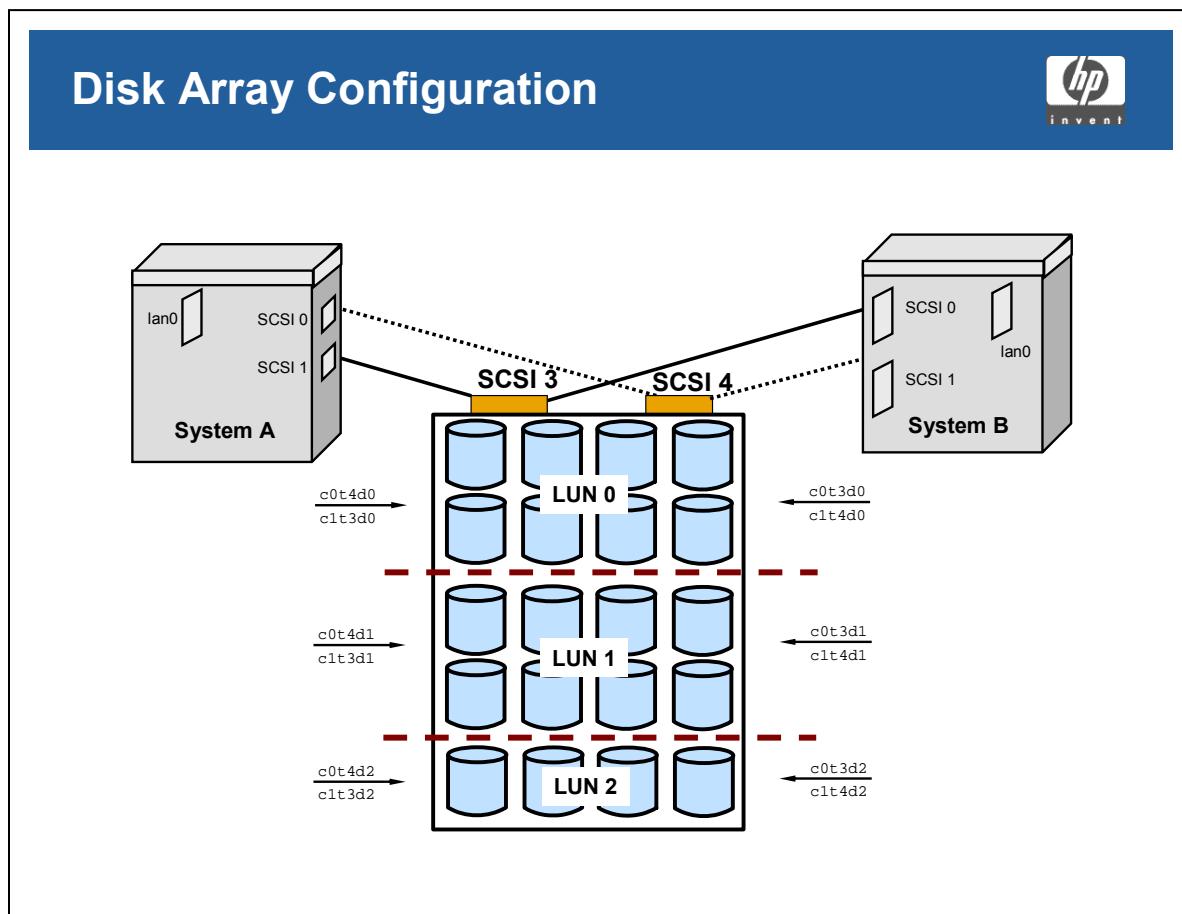


### Student Notes

The disk technology that uses multiple standalone disk drives is also known as *just a bunch of disks* (JBODs).

High Availability Disk Enclosures can rack-mount multiple disk drives in storage cabinets, each disk drive being hot-pluggable. Every component is highly available (dual components) including the power supplies, power cords, cooling fans, and internal Fibre Channel or SCSI buses.

### 3-8. SLIDE: Disk Array Configuration



### Student Notes

#### High Availability Disk Arrays

High availability disk arrays are only supported in a Serviceguard environment in which dual service processors (SPs) are used. Each SP is assigned a SCSI address and appears to the HP-UX operating system as one large disk (or several large disks using SCSI LUNs). Only the SPs are assigned SCSI addresses; the individual disk drives are not.

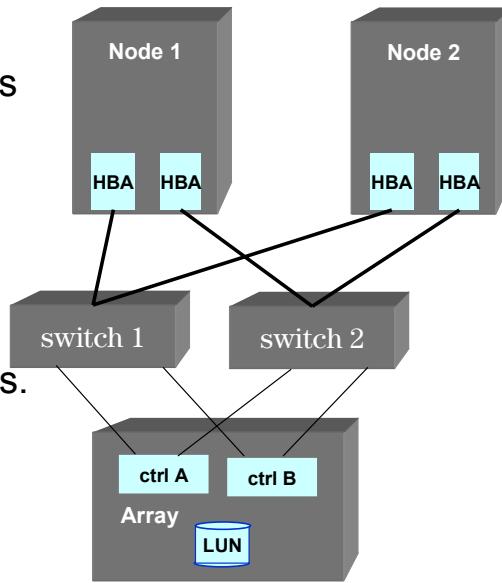
If SCSI logical unit numbers (LUNs) are used, each SCSI LUN will reference one or more disk drives within the array. Within each LUN, a different RAID level can be defined. The example on the slide shows three LUNs, with two LUNs containing eight disks and one LUN containing four disks. All of the LUNs can be configured to be accessible by both SPs, providing high availability at the SP level.

### 3-9. SLIDE: Highly Available SANs

## Highly Available SANs



- Storage Area Networks make it easy to share disks between multiple systems
- Avoid all SPOFs in a SAN and have redundant switches
- The LUN will be visible to the hosts via multiple paths. You must choose a multi-pathing solution



### Student Notes

Storage Area Networks (SANs) make it easier to share storage between multiple systems. A separate network is created for the storage. LUNs can be made visible to any hosts connected to the SAN. As with other types of disk configurations, single points of failure must be avoided.

### Multi-pathing Solutions

In the example on the slide each host will see the LUN twice. To HP-UX the one LUN will look like two different devices. A multi-pathing solution must be used in this situation.

One choice is to use LVM PVlinks. This is a no-cost solution, but one path will be the primary and the other path will only be used in case of a failure. Also, you must manually configure both paths.

Another choice is to purchase multi-pathing software such as Veritas Dynamic Multi-Pathing (DMP), EMC Power-path, or HP Secure Path. All of these products support load balancing across the paths and have the ability to dynamically discover the multiple paths.

## 3-10. SLIDE: Network Configurations

### Network Configurations



#### Possible Network Configurations for a Serviceguard Cluster Include:

- Redundant LANs
- Normal Network Flow
- Network Flow during Package Failover
- Routed Networks and Serviceguard Clusters

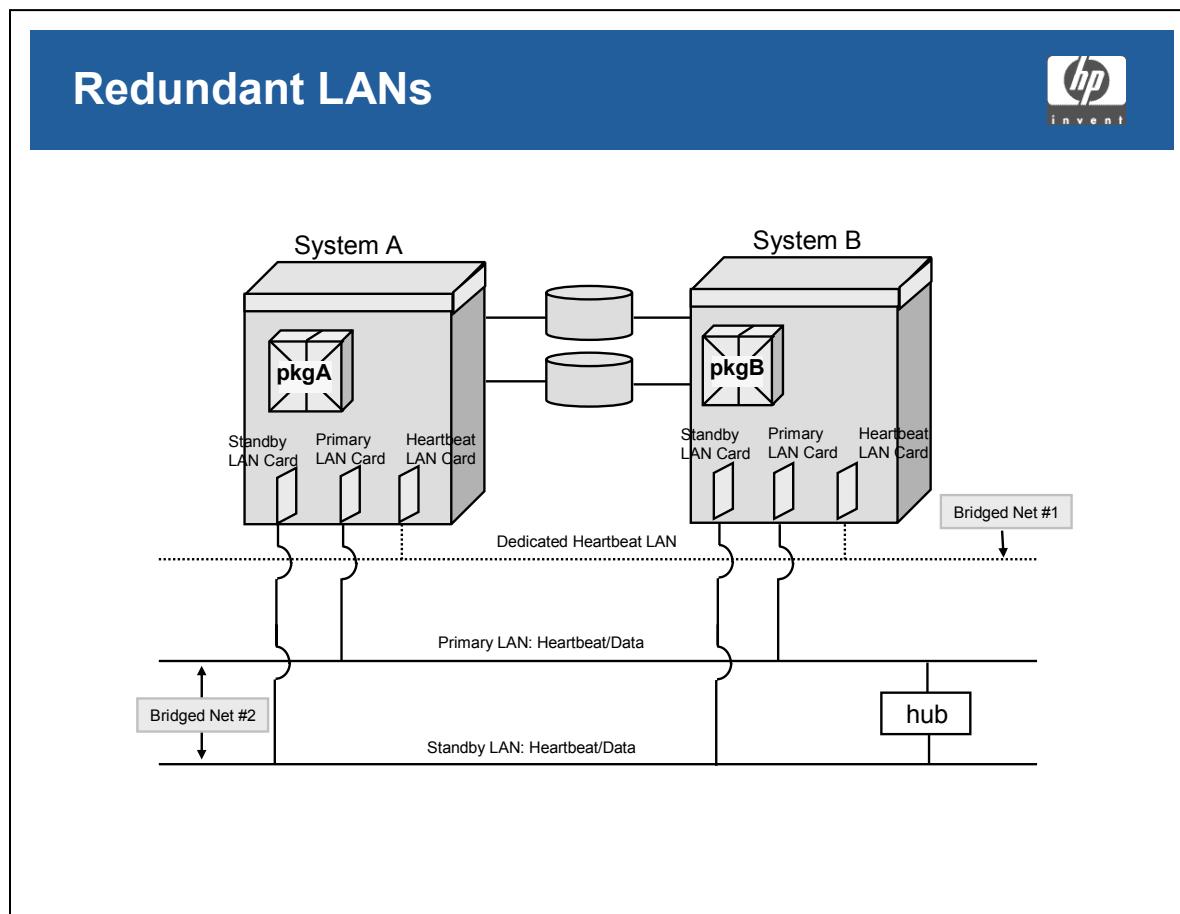
### Student Notes

To eliminate single points of failure for networking, redundant network interfaces, redundant cables, and redundant LANs are required.

There are many possible network configurations to achieve network redundancy. Some of the configurations are supported, and others are not supported in a Serviceguard environment.

The following slides provide a sample of supported and unsupported network configurations for a Serviceguard environment.

### 3-11. SLIDE: Redundant LANs



### Student Notes

An example of redundant LAN interfaces to the same network is shown on the above slide. In the slide, each Serviceguard node is connected to two "bridged networks." The first connection is a "dedicated" connection to a dedicated Heartbeat LAN subnet. This is a standard type connection with no redundant LAN connections. If the heartbeat LAN card fails, then the node will be unable to get the heartbeat LAN subnet.

The second connection is to a Heartbeat/DATA subnet. In this case, each node has two connections to the subnet: a primary LAN card connection and a standby LAN card connection. Note the primary and standby LAN segments are connected by a switch (it could also be a hub or a bridge) to provide redundancy between the two segments of the Heartbeat/DATA subnet. In the case of a primary LAN card failure for the Heartbeat/DATA subnet, Serviceguard will perform a local switch of the IP addresses on the primary LAN card to the standby LAN card.

The reason for not having a standby LAN card for the dedicated Heartbeat LAN: redundancy is already being provided by sending heartbeat packets across both LAN subnets. If a LAN card to the dedicated heartbeat LAN subnet is lost, the cluster can continue as long as the heartbeat can get through to each node on the Heartbeat/DATA subnet. Serviceguard only requires one of the heartbeat packets to get through, not both.

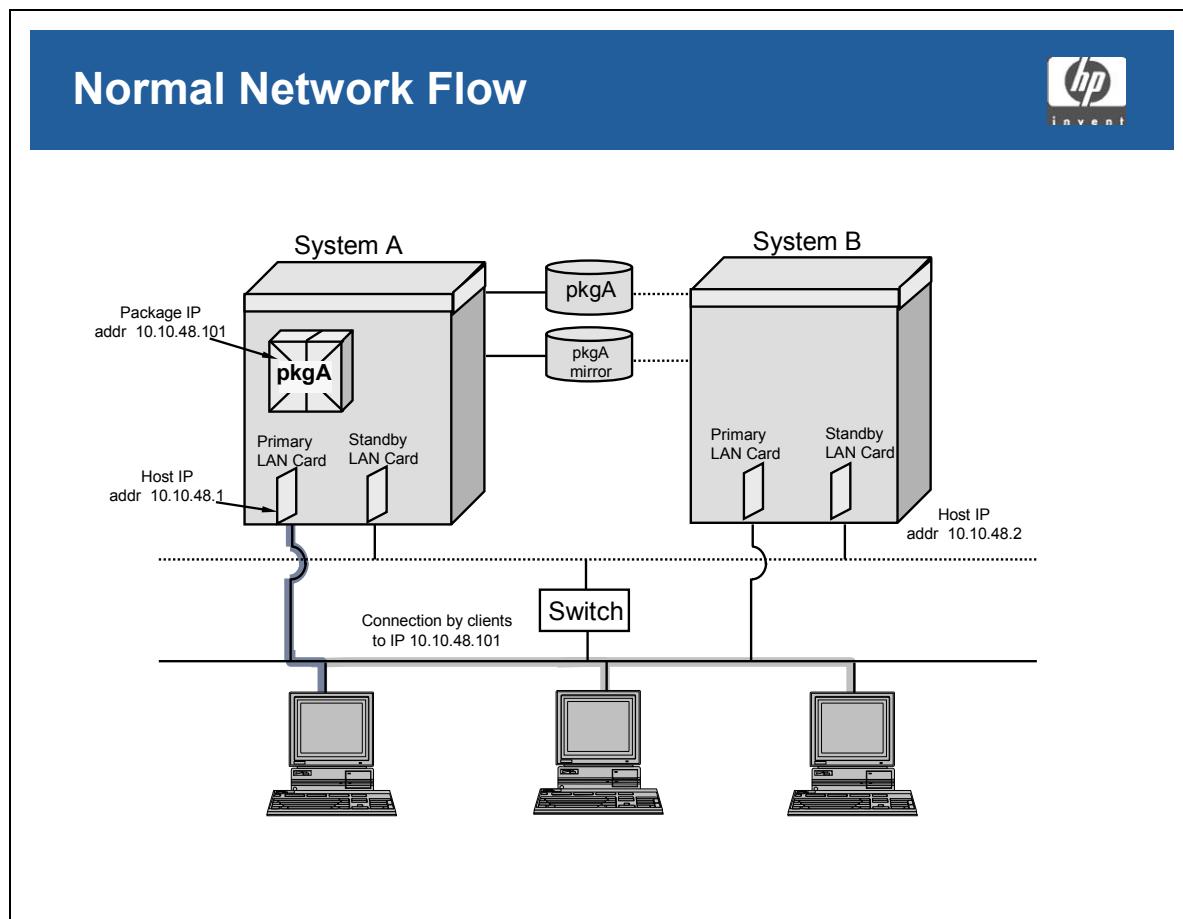
## Standby LAN Cards

Standby LAN cards are configured automatically by Serviceguard during the cluster setup/configuration phase. The two requirements to be a standby LAN card are:

- The LAN card must be unconfigured (it must not have an IP address assigned).
  - The standby LAN card must be on the same physical network as the primary LAN card. In other words, the standby LAN card must be able to communicate (send packets) to the same nodes on which the primary LAN card communicates.

The configuration shown on the slide is a supported network configuration for a Serviceguard environment. The bridge is a key component, because it combines the two physical LAN segments into one continuous bridged network.

### 3-12. SLIDE: Normal Network Flow



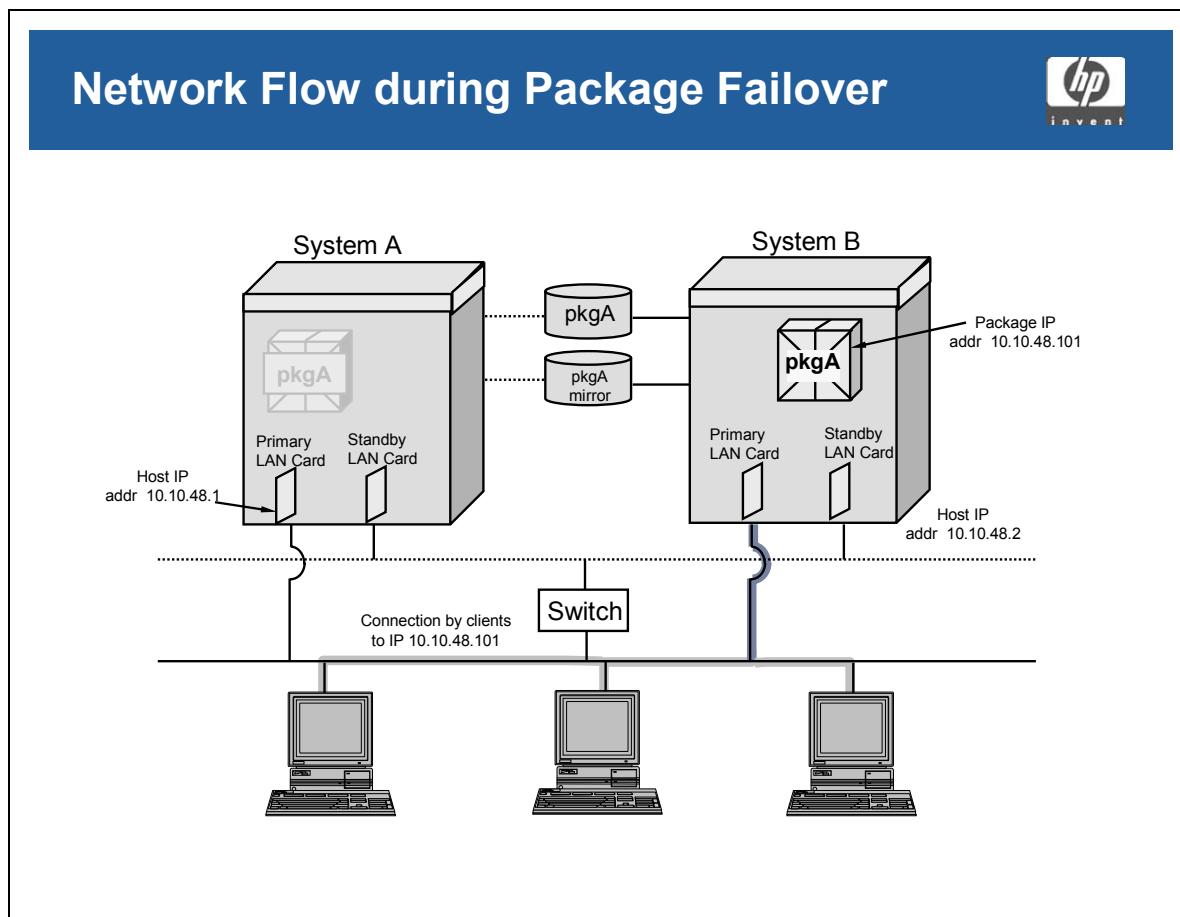
### Student Notes

The above slide shows the how clients connect to HA applications in a Serviceguard environment. From the outside, it *appears* as the client connects with the server hosting the HA application (as normal). But, internally, something quite different is happening.

While SystemA still contains an IP address that represents the host computer, it also contains a "floating" IP address that represents the application. Recall that all resources associated with the application are combined into an entity called a "package." When an application starts up, one of the resources for the package, namely the IP address, is added to the LAN card of the host on which it is running. Therefore, once the application starts, there are two IP addresses associated with the primary LAN card on SystemA: the IP address of the system itself (10.10.48.1) and the IP address of the application (10.10.48.101).

When a client connects to the application in a Serviceguard environment, it does *not* connect with the IP address of the system hosting the application; instead, it connects with the IP address of the application itself! In the above slide, notice the clients are connected to IP address 10.10.48.101, which is the IP address of PackageA.

### 3-13. SLIDE: Network Flow during Package Failover



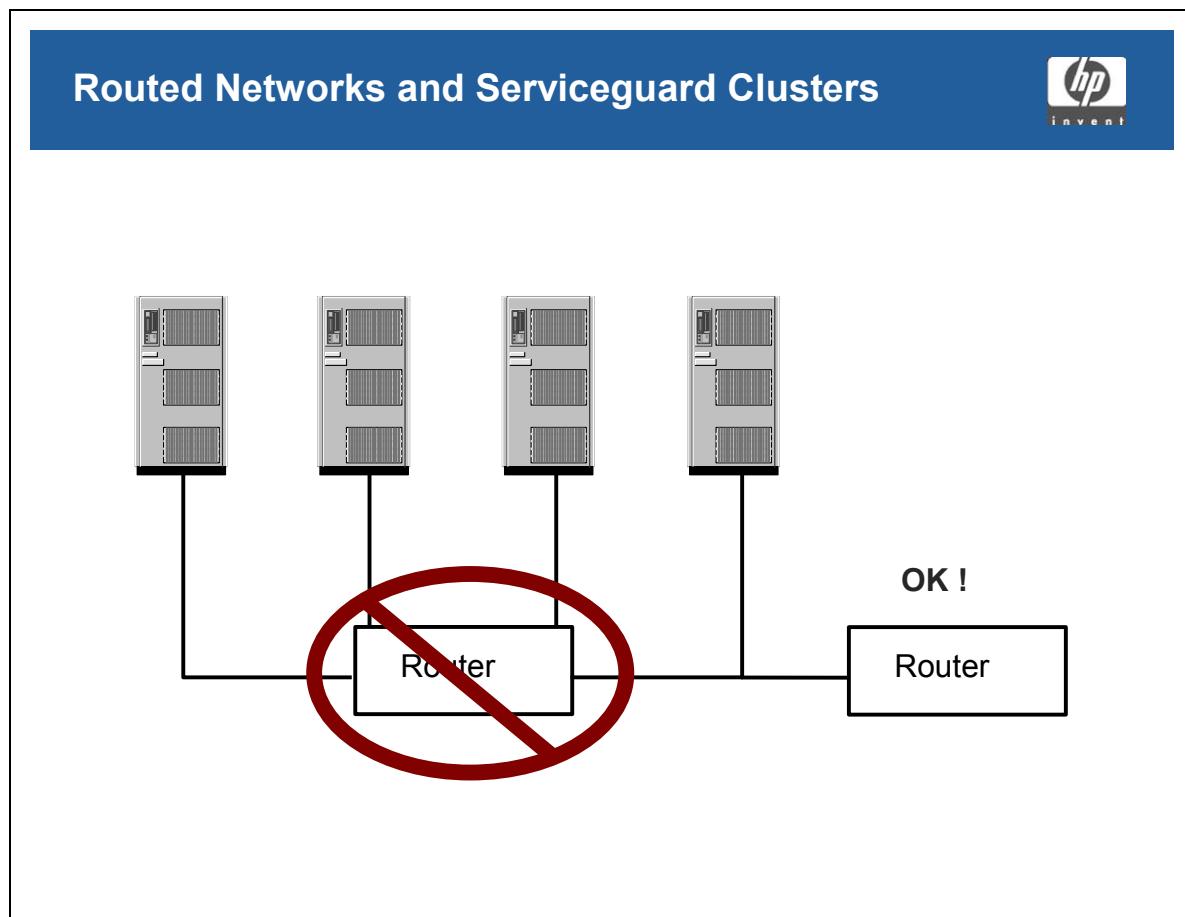
#### Student Notes

The above slide shows the impact on the network flow when a package moves from one host to another host in a cluster.

The main impact of moving a package is the IP address of the application moves when the package moves. On the previous slide, we saw the clients were sending their network packets to SystemA, because that was where the IP address of PackageA was located. During a package failure or package move, the IP address of the application is de-configured off the primary system's LAN card, and is configured onto the secondary system's LAN card. As a result, all network packets destined to the IP address of the application are now routed to the secondary system, since that is the new location of the package's IP address.

The major benefit of having clients connect to the "floating" IP address of the application is the clients do not need to be re-configured when the application moves. If there were no IP address associated with the application, then the clients would all need to be reconfigured when the application started on a new node. But, because Serviceguard supports floating IP addresses, client modifications are not needed when an application moves to another node in the cluster.

### 3-14. SLIDE: Routed Networks and Serviceguard Clusters

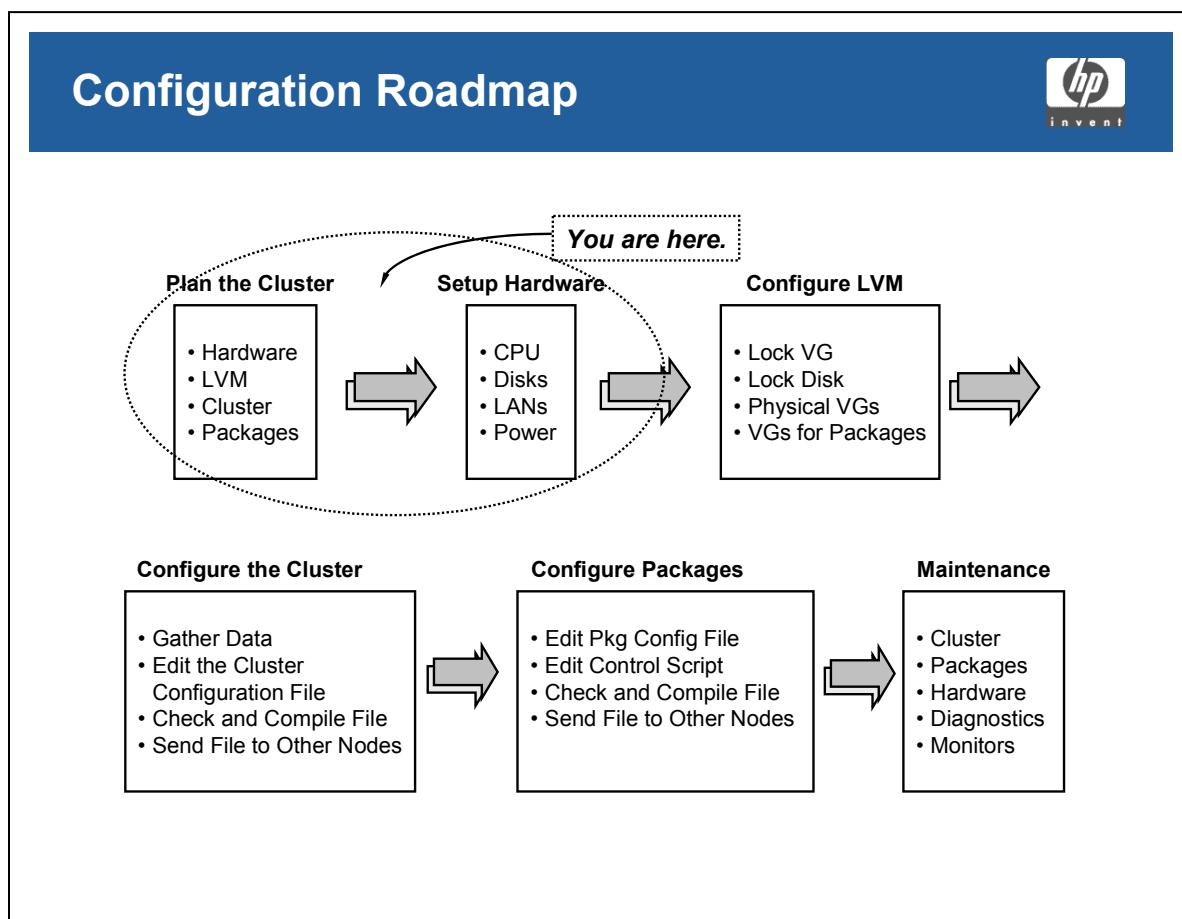


#### Student Notes

There are a number of rules to remember when working with routers in a Serviceguard environment:

- The members of an HA Cluster may not be connected to each other via a router. All nodes must be on the same segment in a routed network. Therefore, the Enterprise Cluster can connect to a routed network.
- Although the heartbeat uses IP, cluster formation uses link level addressing (streams DLPI- Data Link Provider Interface). The complicated "state machine" and timing concerns at cluster formation means that a short deterministic algorithm must be used. Sockets, which are associated with higher-level protocols, would take too long to perform this function. Therefore, heartbeats do not communicate via sockets.
- These requirements also prevent HA Clusters from working in a WAN environment.

### 3-15. SLIDE: Configuration Roadmap



#### Student Notes

The above slide provides a good "roadmap" of activities needed to configure and maintain a Serviceguard environment.

At this point in the class, you are about to complete the "Plan the Cluster" and "Setup Hardware" phases by performing the "Hardware Inventory" lab exercise at the end of this module.

The next steps, as shown on the slide, will be to:

- Configure a share volume group using LVM
- Configure/Setup the Cluster
- Configure Package to execute in the Cluster
- Perform Cluster and Package maintenance activities.

## 3-16. LAB: Hardware Inventory

### Directions

Planning is an essential component of any successful Serviceguard implementation. Part of that planning involves identifying all hardware that will be used in the cluster and identifying what it will be used for. This includes servers, LAN cards, and disks.

In this lab, we will examine some of the hardware requirements of Serviceguard and identify the servers, LAN cards, and disks we will be using in our classroom implementation of a Serviceguard Cluster. We will also mirror the root disk.

### Part 1: Identify the Servers

Data to Record	Command Used	Server1	Server2
Node Name	hostname uname -a		
System Model	model		
OS Rev	uname -r		
Memory	dmesg		
Date / Time / Timezone	date		
Serviceguard Revision	swlist -l product \   grep -i guard		

---

**NOTE:** The servers do not need to be identical in a Serviceguard cluster.  
Packages containing applications will failover more smoothly if date and time are synchronized and if the adoptive node has adequate resources (CPU, memory, etc.) to run the application.

---

## **Part 2: Identify the LAN Cards**

Each server in a Serviceguard cluster must have at least two LAN cards.

In a production environment, HP strongly recommends at least three LAN cards per server with at least one LAN card per server used exclusively for heartbeats (as a dedicated heartbeat LAN).

In our classroom cluster, we will not have a separate, dedicated heartbeat LAN card on each server. Instead, each server will have a primary LAN card and a standby LAN card. Some network traffic will use the public LAN cards. Other network traffic (heartbeats and client/server data) will use the private LAN cards (to be configured later). If a LAN card fails, all traffic will be sent via the standby LAN card.

The primary LAN cards must be configured with an IP address. The standby LAN cards must not have an IP address. Serviceguard will automatically assign the primary's IP address to the standby if the primary fails. The primary and standby LAN cards on both servers must all reside on the same bridged-subnet. Let's verify all of this.

<b>First, consider the Primary LAN Card</b>	<b>Command</b>	<b>Server1</b>	<b>Server2</b>
IP Address	<code>ifconfig lanX     ( often lan0 ) netstat -in</code>		
Subnet	<code>netstat -in</code>		
Hardware Path	<code>lanscan ioscan -fc lan</code>		
Station Address	<code>lanscan</code>		
Net-Interface Name / PPA	<code>lanscan</code>		
Description ( from ioscan )	<code>ioscan -fc lan</code>		

Module 3  
**High Availability Planning**

<b>Now, consider any Standby LAN Card</b>	<b>Command</b>	<b>Server1</b>	<b>Server2</b>
IP Address	ifconfig lanY ( often lan1 ) netstat -in	These LAN cards <b>must not</b> have an IP address assigned.	
HW Path	lanscan ioscan -fC lan		
Station Address	lanscan		
Net-Interface Name / PPA	lanscan		
Description ( from <b>ioscan</b> )	ioscan -fC lan		

1. If you are using local equipment, make certain you know which LAN card is in which slot in the backplane of each server. If you are using remote equipment, continue with the next step.
2. Are all LAN cards physically attached to the same bridged-subnet? Let's test.

Using the `linkloop` command, verify that, on each server, you have connectivity between all of the available `lan` cards on that server **AND** on the other server.

---

**NOTE:** The `linkloop` command can test connectivity of a LAN card that has no IP address. `linkloop` takes the station address as an argument.

---



---

**NOTE:** The `linkloop` command, without options, assumes you intend to send a test packet out of the first LANIC in the system. Therefore, if you are using another LANIC as the origination LANIC, it will be necessary to use the `-i` option with the appropriate interface. The default for the `-i` option is `-i 0`. Thus, if your currently configured system LANIC is `lan1`, then you would enter:

---

```
# linkloop -i 1 0x_destination_station_address
```

---

## On Server 1:

```
# linkloop [ -i N ] 0x_1st_lan_card_server1_station_address
# linkloop [ -i N ] 0x_2nd_lan_card_server1_station_address
# linkloop [ -i N ] 0x_3rd_lan_card_server1_station_address
# linkloop [ -i N ] 0x_4th_lan_card_server1_station_address

# linkloop [ -i N ] 0x_1st_lan_card_server2_station_address
# linkloop [ -i N ] 0x_2nd_lan_card_server2_station_address
# linkloop [ -i N ] 0x_3rd_lan_card_server2_station_address
# linkloop [ -i N ] 0x_4th_lan_card_server2_station_address
```

### On Server 2:

```
# linkloop [ -i N ] 0x_1st_lan_card_server2_station_address  
# linkloop [ -i N ] 0x_2nd_lan_card_server2_station_address  
# linkloop [ -i N ] 0x_3rd_lan_card_server2_station_address  
# linkloop [ -i N ] 0x_4th_lan_card_server2_station_address  
  
# linkloop [ -i N ] 0x_1st_lan_card_server1_station_address  
# linkloop [ -i N ] 0x_2nd_lan_card_server1_station_address  
# linkloop [ -i N ] 0x_3rd_lan_card_server1_station_address  
# linkloop [ -i N ] 0x_4th_lan_card_server1_station_address
```

If any `linkloop` command fails, we must correct the LAN connectivity problem before proceeding with building the cluster.

### **Answer:**

All LAN cards should be able to communicate with each other. If there are connectivity issues, notify your instructor.

### Part 3: Identify the Disks

Recorded Data	Command	Server1	Server2
Number of (a) sharable SCSI cards or (b) sharable bus adapters	(a) ioscan -fC ctl or (b) ioscan -fC ba		
Hardware Path of (a) sharable SCSI cards or (b) sharable bus adapters ( e.g. 8/0 )	(a) ioscan -fC ext_bus or (b) ioscan -fC ba		
Initiators  Hardware address of (a) the SCSI card itself, or (b) hardware address of sharable bus adapters ( e.g. 8/0.7.0 )	(a) ioscan -fC ctl or (b) ioscan -fC ext_bus		
Hardware Paths of sharable Disks ( e.g. 8/0.5.0 )	ioscan -fC disk		
Device file names of sharable disks ( e.g. c0t5d0 )	ioscan -fnC disk		

1. If you are using local equipment, determine which disks are private and which are shared (you may be able to trace the cables). If you are using remote equipment, ask your instructor for this information
  2. **Special note:** If you are using local equipment, then continue with this question. If you are using remote equipment, then skip this question completely, and go immediately to question 4 below.

Identify the physical location of each specific disk. How do you know which disk is in which slot in which cabinet?

**Answer:**

A simple way of locating a disk is to issue the following command, and watch which disk's access light illuminates.

```
# dd if=/dev/rdsk/disk in question of=/dev/null
```

- ### 3. What volume groups currently exist?

**Answer:**

```
# strings /etc/lvmtab
```

4. Which disks are currently in use and owned by these existing volume groups?

## Answer:

```
# vgdisplay -v  
# strings /etc/lvmtab
```

**Informational only:** Does each shared disk have the same device file name regardless of which server we use to access the disk? Would it be possible for the same disk to have a different device file name on `server1` vs. `server2`? Would Serviceguard work in this environment?

When the nodes of a cluster are all identical in their configurations (including device file names for shared disks), this cluster is referred to as a "Symmetric Cluster."

For more on maintaining symmetric clusters, please continue with Part 4 on the next page.

## Part 4: Information on Maintaining Symmetric Clusters

Note: This section of the lab is informational only. There is no work that must be completed in this section of the lab.

If a disk is simultaneously attached to SCSI card instance 1 on server1 and SCSI card instance 2 on server2, the disk will have a different device file name on server1 vs. server2. While Serviceguard will work in this environment, it may be less confusing if this situation is avoided.

Most Serviceguard administrators would prefer a cluster that is symmetric (where the disk device files for shared volume groups are identical between systems) as it is easier to administer. And while it is possible to alter device file names through the procedure below, this concept is quickly becoming a moot point with the introduction of SAN's and other disk arrays into Serviceguard environments. With SAN's and certain arrays, it is nearly impossible to create identical disk device file names between systems.

In a SCSI disk environment, it is possible to alter instance numbers in order to create identical device file names for disks. The procedure follows:

Begin by creating a file that maps H/W addresses to instance numbers and then run **ioinit** against this file.

For example, if the current instance number assignment on server1 looks like this:

8/0	ext_bus	0
8/4	ext_bus	1
8/8	ext_bus	2

and the current instance number assignment on server2 looks like this:

8/0	ext_bus	2
8/4	ext_bus	1
8/8	ext_bus	0

we can use the following method to make server2 look like server1:

To give the 8/0 SCSI card an instance of 0, and the 8/8 SCSI card an instance of 2, create a file called **/tmp/mappings** which contains:

8/0	ext_bus	0
8/4	ext_bus	1
8/8	ext_bus	2

Then, execute **ioinit -f /tmp/mappings -r** on server2. The system will reboot with the new mapping. This must be done **BEFORE** the shared volume groups are created!

---

### 3-17. LAB: Configuring and Managing LVM Boot Disks on Integrity-Based Systems (Optional)

#### Special Note

- a. Lab 3-17 is for those using Integrity, or Itanium-based, systems.
- b. If you are using PA-RISC systems, skip this lab and proceed to Lab 3-18.
- c. If you are not sure which equipment you have, ask your instructor.

## Overview

In this hands-on lab, you will create mirrors for all of the logical volumes that comprise the root volume group, vg00. **These steps should be performed on both systems in the cluster to improve the availability of the root disk on each system of the cluster.**

**Carefully read and follow the instructions below.**

## Part 1: Mirroring an Integrity Boot Disks: Configuring Partitions

In order to improve system stability, many administrators choose to mirror the system boot disk. Doing so guarantees that your system continues running, even if one of the boot disks or SCSI interface cards on your system is damaged.

1. First, use the `lvolnboot` command to determine the location of your current boot disk. After that, use the `ioscan` and `diskinfo` commands to identify a disk large enough to serve as a mirror.

```
# lvolnboot -v
# ioscan -func disk
# diskinfo /dev/rdsck/cxtwdx
```

2. Use the `idisk` command to create a 500 MB system partition, an OS partition, and a 400 MB service partition on your new boot disk.

```
# vi /tmp/idf
```

```
3
EFI 500MB
HPUX 100%
HPSP 400MB
```

```
# idisk -wf /tmp/idf /dev/rdsk/cxtxdx
```

Note that the screen output from this `idisk` command is fairly large. Down about 21 lines from the top of the screen output, you should see something like the following, proving that the three partitions have been created:

```
Primary Partition Table (in 512 byte blocks):
Partition 1 (EFI):
    Partition Type GUID      = c12a7328-f81f-11d2-ba4b-00a0c93ec93b
    Unique Partition GUID    = cca5f42a-96e8-11d9-8002-d6217b60e588
    Starting Lba            = 0x22
    Ending Lba              = 0xfa021
Partition 2 (HP-UX):
    Partition Type GUID      = 75894c1e-3aeb-11d3-b7c1-7b03a0000000
    Unique Partition GUID    = cca5f448-96e8-11d9-8003-d6217b60e588
    Starting Lba            = 0xfa022
    Ending Lba              = 0x430e6fb
Partition 3 (HPSP):
    Partition Type GUID      = e2a1e728-32e3-11d6-a682-7b03a0000000
    Unique Partition GUID    = cca5f466-96e8-11d9-8004-d6217b60e588
    Starting Lba            = 0x430e6fc
    Ending Lba              = 0x43d66fb

Legacy MBR (MBR Signatures in little endian):
    MBR Signature = 0x46f1a5cc

Protective MBR
```

3. Create device files for the new partitions.

```
# insf -eC disk
```

Module 3  
**High Availability Planning**

4. Use the `idisk` command again, this time to verify the partition table on the new boot disk.

```
# idisk /dev/rdsk/cxtxdx
```

Once again, in the middle of the `idisk` screen output, you should see three partitions.

5. Verify that the partition device files were created properly. There should be block **and** raw device files (`cxtxdxs1`, `cxtxdxs2`, and `cxtxdxs3`) for the new boot mirror.

```
# ioscan -func disk
```

## Part 2: Mirroring the Boot Disk: Configuring the System Partition

After you create the partition table, you can initialize and populate the EFI system partition.

1. First, populate the `/efi/hpx` directory in the new EFI partition.

```
# mkboot -e -l /dev/rdsck/cxtwdx      ← Takes about 5 seconds to execute
```

2. Change the `auto` file **on both boot disks** so the host can boot without quorum. Change the boot string in the file to `boot vmunix -lq`. Be sure to specify section `s1`, the EFI partition in your `efi_cp` command! Begin by creating a temp file to store the boot string.

```
# vi /tmp/auto
```

```
boot vmunix -lq
```

```
# efi_cp -d /dev/rdsck/cxtwdxsl /tmp/auto /efi/hpx/auto
# efi_cp -d /dev/rdsck/cytydys1 /tmp/auto /efi/hpx/auto
```

3. Verify the contents of the system partition. Again, be sure to specify the `s1` partition!

```
# efi_ls -d /dev/rdsck/cxtwdxsl
```

4. Verify the contents of the `auto` file in the system partition.

```
# efi_cp -d /dev/rdsck/cxtwdxsl -u /efi/hpx/auto /tmp/auto.contents
# cat /tmp/auto.contents
```

## Part 3: Mirroring an 11i v2 Boot Disk: Configuring the OS Partition

Now, configure the OS partition. Recall that the OS partition is essentially an encapsulated LVM boot disk.

1. Initialize and add the new OS partition to vg00. Be sure to specify section `s2`, the OS partition!

```
# pvcreate -fB /dev/rdsk/cxtxdxs2
# vgextend vg00 /dev/dsk/cxtxdxs2
# lvlnboot -v
```

← To see your newly initialized (though still incomplete) boot disk.

2. Mirror *all* of the logical volumes in vg00. Depending on the number, and size, of your logical volumes in vg00, this step could take up to 15 minutes to complete.

```
# for i in /dev/vg00/lvol*
> do
> echo "Now mirroring $i ...\\n\\n"
> lvextend -m 1 $i
> done
```

3. Be sure to add a new line to the `/stand/bootconf` file so that SDUX knows which disks are boot disks. Be sure to specify section `s2`, the OS partition!

```
# vi /stand/bootconf
```

1 /dev/dsk/cxtxdxs2	← The first character in this line is an “ell”, not a “one”
1 /dev/dsk/cytydys2	← The first character in this line is an “ell”, not a “one”

4. Verify that the disk was added to `vg00`, and that the logical volumes are synchronized. For each logical volume, the LV Status field should report “available/syncd”.

```
# vgdisplay -v vg00 | more
```

5. Verify that the BDRA was updated properly. You should see references to both disks in the boot, root, and swap portions (but not the dump portion) of the output.

```
# lvlnboot -v
```

## **Part 4: Configuring an HP-UX 11i v2 Boot Disk: Configuring the EFI Boot Manager Menu**

Finally, add the new mirrored disk to the EFI boot manager menu. This ensures that the EFI will automatically attempt to boot the mirrored boot disk if the primary boot disk fails.

1. Ensure that your initial boot disk is configured as your primary boot source, and add your new boot disk to the EFI boot manager as the second (i.e. high availability alternate) boot source.

**If needed, replace the hardware paths below** with the hardware paths for your first and second boot disks.

```
# setboot                                ← To see the current configuration
# setboot -p 0/1/1/0.1.0                  ← If needed, to set the primary boot path
# ioscan -funC disk                      ← Obtain hardware path for the mirror disk
# setboot -h 0/1/1/1.2.0                  ← Set the "High Availability Alternate"
                                            (or "HAA") boot path.
```

2. Verify that your new boot path was recorded properly in NVRAM.

```
# setboot
```

## Part 5: Mirroring the Boot Disk: Testing the New Mirror

Now for the real test: Can you successfully boot from the alternate disk? Try it!

### Special Note

- a. If you are using remote equipment, remember to log into Unix using the GSP/MP. That way, you will be able to watch the boot sequence. You will also be prepared to press the “down arrow” key mentioned in step 1 below.
- b. If you are using local equipment, and are logged in to Unix directly on the console, proceed immediately with step 1 below.

1. Use the `shutdown` command to initiate the boot sequence. When given an opportunity to interrupt the EFI autoboot sequence, do so by pressing the down arrow key.

```
# shutdown -ry 0
```

When the EFI Boot Manager menu appears, press the “down arrow” key.

2. Select your mirrored disk from the boot manager selection menu and proceed with the boot process.

```
EFI Boot Manager ver 1.10 [14.60] Firmware ver 1.61 [4241]
Please select a boot option
    HP-UX Primary Boot
    HP-UX HA Alternate Boot
    EFI Shell [Built-in]
```

3. When the boot process finishes, check the `syslog.log` file to verify which disk/kernel you booted from. You should see something like the following:

```
# grep "Boot device's HP-UX HW path" /var/adm/syslog/syslog.log
vmunix: Boot device's HP-UX HW path is: 0.0.0.0.1.0
```

4. Congratulations on mirroring your boot disk on an Integrity server! At this point, you can either stay booted from your mirror copy, or do a `shutdown -ry 0` and let the system auto-boot using the primary copy.

```
# shutdown -ry 0      ← Optional
```

## 3-18. LAB: Configuring and Managing LVM Boot Disks on PA-RISC-Based Systems (Optional)

### Special Note

- a. Lab 3-18 is for those using PA-RISC systems.
- b. If you are not sure which equipment you have, ask your instructor.

## Overview

In this hands-on lab, you will create mirrors for all of the logical volumes that comprise the root volume group, vg00. **These steps should be performed on both systems in the cluster to improve the availability of the root disk.**

**Carefully read and follow the instructions below. Perform the following steps on each node that will be used for Serviceguard:**

## Directions

1. First, find the names of the disks you will use. These instructions will use the following disks:

/dev/dsk/c0t5d0	( already part of vg00, by default )
/dev/dsk/c0t8d0	( second internal disk on the same controller )

If the systems you are using have different device file names, then please change the above device file names accordingly.

2. Ensure that the second internal disk is available for use in this mirroring lab.

```
# strings /etc/lvmtab
```

If this disk (/dev/dsk/c0t8d0) is not available, please notify your instructor.

3. Now, initialize a disk as a boot device for use with LVM (be sure to use the raw device file).

```
# pvcreate -B [ -f ] /dev/rdsk/c0t8d0
```

4. Extend the vg00 volume group to include the new disk (use the block device file).

```
# vgextend vg00 /dev/dsk/c0t8d0
```

5. Now, let's check the disk space on the newly added disk to verify that there is sufficient space to add the boot programs to the disk. If the disk at `/dev/dsk/c0t8d0` has less space than the disk at `/dev/dsk/c0t5d0`, ask your instructor for assistance.

```
# diskinfo /dev/dsk/c0t5d0
# diskinfo /dev/dsk/c0t8d0
```

6. Copy the boot programs to the LVM disk.

```
# mkboot /dev/rdsck/c0t8d0
```

7. Verify that the LIF programs were copied to the disk.

```
# lifls -l /dev/rdsck/c0t8d0
```

8. Verify the LVM configuration on the root volume group.

```
# lvlnboot -v
```

9. Using the following script, mirror each of the logical volumes, in order, onto the second disk. Depending on the number, and size, of your logical volumes in `vg00`, this step could take up to 15 minutes to complete.

```
# for i in /dev/vg00/lvol*
> do
> echo "Now mirroring $i ...\\n\\n"
> lvextend -m 1 $i
> done
```

10. Verify that the mirrored volumes are registered correctly in the `BDRA` for `vg00`.

```
# lvlnboot -v
```

11. Now, add a new line to the `/stand/bootconf` file so that SDUX knows which disks are boot disks.

```
# vi /stand/bootconf
```

1 /dev/dsk/cxtxdx ← The first character in this line is an “ell”, not a “one”  
1 /dev/dsk/cytydy ← The first character in this line is an “ell”, not a “one”

12. Display the contents of the LIF file named AUTO, which is the "Autoexecute" file used by the HPUX secondary loader at boot time. This file normally contains the "hpx" startup string. The command shown below copies the AUTO file to STDOUT.

```
# lifcp /dev/rdsk/c0t5d0: AUTO -
```

13. Modify the `AUTO` file to allow the system to boot without LVM quorum. This should be performed on both boot disks in the root volume group.

```
# mkboot -a "hpx -lq" /dev/rdsk/c0t5d0
```

```
# mkboot -a "hpx -lq" /dev/rdsk/c0t8d0
```

14. Check the primary and alternate boot paths in use on the system.

```
# setboot
```

15. Finally, change the alternate boot path to be the second disk, if the current value is something other than the second disk.

```
# setboot -a 8/0.8.0
```

Module 3  
**High Availability Planning**

16. If you have time available, you may test your mirroring by ...

- rebooting the system
- interrupt the boot sequence ( "press any key" )
- SEArch for potential boot devices
- boot from the mirror copy

---

## **Module 4 — LVM for Serviceguard**

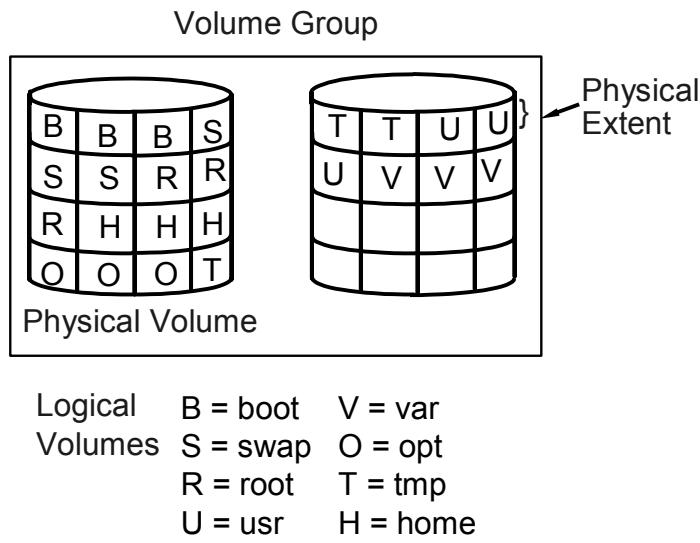
### **Objectives**

Upon completion of this module, you will be able to do the following:

- Understand how to configure a shared volume group for a Serviceguard environment.
- Understand how to export and import volume groups without having to specify disk devices.
- Understand how to activate a shared volume group in exclusive read/write access mode.
- Understand the list of operating system files that need to be modified to support shared volume groups in a Serviceguard environment.

## 4-1. SLIDE: Review of LVM Concepts

### Review of LVM Concepts



### Student Notes

Logical Volume Manager ( LVM ) is the default disk management subsystem replacing the traditional hard disk sectioning used on HP 9000 systems. It allows the system manager much greater flexibility in controlling the size of the various disk partitions or logical volumes. As with traditional disk sections, logical volumes can be used for file systems, raw data, swap locations, or dump locations. A logical volume can be used in place of a disk section or a whole disk in an HP-UX administration command.

### LVM Terminology

There are four basic LVM terms upon which this module is based: volume groups, physical volumes, logical volumes, and physical extents.

#### Volume Groups

The volume group is the most fundamental concept in LVM. In simple terms, it is a group of disk drives (also called physical volumes). The volume group is comprised of all the space on all the disks. When space is allocated from the volume group, it is done without concern for which disk the space is allocated on.

### **Physical Volumes**

The physical volume is the physical disk drive. When configuring a disk for use under LVM, the entire disk must be allocated to a volume group. Once the disk is in the volume group, we refer to the disk as a physical volume, indicating the disk is under the control of LVM.

### **Logical Volumes**

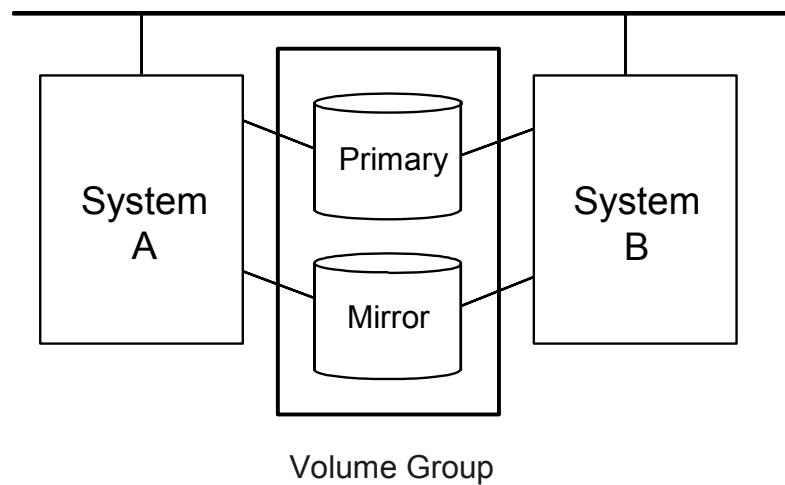
The logical volume is the unit of space which behaves like a disk section. A logical volume can be used to store a file system or it can be referenced as raw disk space. The main difference between logical volumes and disk sections is logical volumes do *not* have to be contiguous; logical volumes can be spread across multiple disks. Logical volumes can be grown dynamically.

### **Physical Extents**

The physical extent is the increment by which space is allocated. All logical volumes and volume groups are measured in physical extents. The default size of a physical extent is 4 megabytes; however, the size can be changed when a volume group is created. The current range is 1–256 megabytes.

## 4-2. SLIDE: Serviceguard Configuration Using Mirrored Disks

### Serviceguard Configuration Using Mirrored Disks



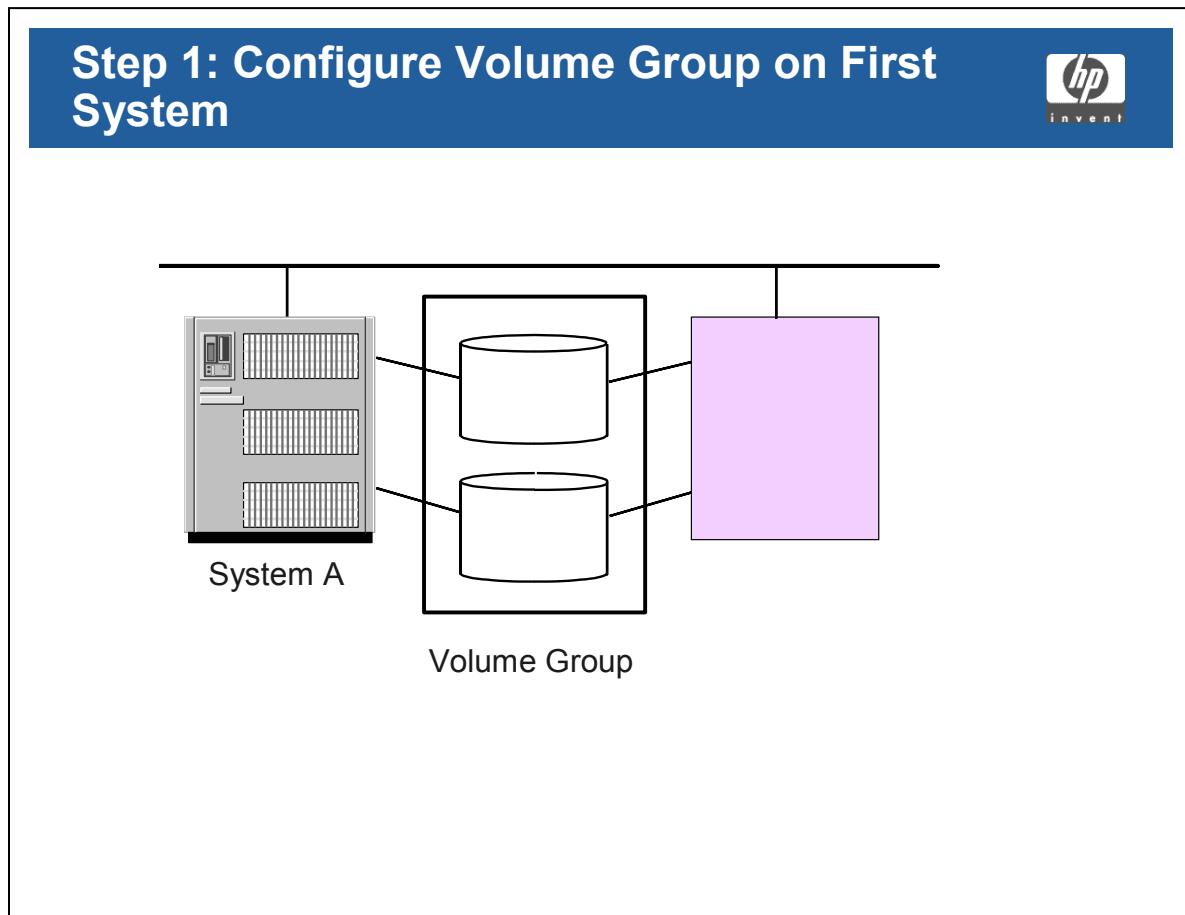
### Student Notes

The slide shows a simple Serviceguard configuration for a two-node cluster. This is the simplest possible configuration for two nodes wishing to operate in a high availability Serviceguard mode.

Notice that the volume group is being shared by system A and system B. Both nodes have equal access to the volume group. The primary disk drive in the volume group is mirrored to provide protection should the primary drive fail, and the mirror is placed on a second SCSI controller card to provide protection in the event of a SCSI controller card failure.

With the above configuration, the system would be able to survive any single failure of a SCSI controller card or a SCSI drive as it relates to the volume group.

### 4-3. SLIDE: Step 1: Configure Volume Group on First System



#### Student Notes

There are two steps when configuring a shared volume group between two systems. The first step is to create the shared volume group on the first system. Examples of the commands to create the shared volume group on the first system are shown below:

#### Set Up the Volume Group

1. Make a directory for the volume group.

```
# mkdir /dev/vg01
```

2. Make the volume group control file (always called “group”).

```
# mknod /dev/vg01/group c 64 0x010000
```

3. Prepare the disks for LVM.

```
# pvcreate [ -f ] /dev/rdsk/c_t_d0
# pvcreate [ -f ] /dev/rdsk/c_t_d0
```

Module 4  
**LVM for Serviceguard**

4. Create the volume group.

```
# vgcreate /dev/vg01 /dev/dsk/c_t_d0 /dev/dsk/c_t_d0
```

5. Once the shared volume group is completed, a mirrored logical volume with a file system can be created in the volume group.

### **Set Up a Mirrored Logical Volume and File System**

1. Create a logical volume in the shared volume group.

```
# lvcreate -L 100 -n mcsq /dev/vg01
```

2. Mirror the logical volume.

```
# lvextend -m 1 /dev/vg01/mcsq
```

3. Create a journaled file system in the logical volume.

```
# newfs -F vxfs /dev/vg01/rmcsq
```

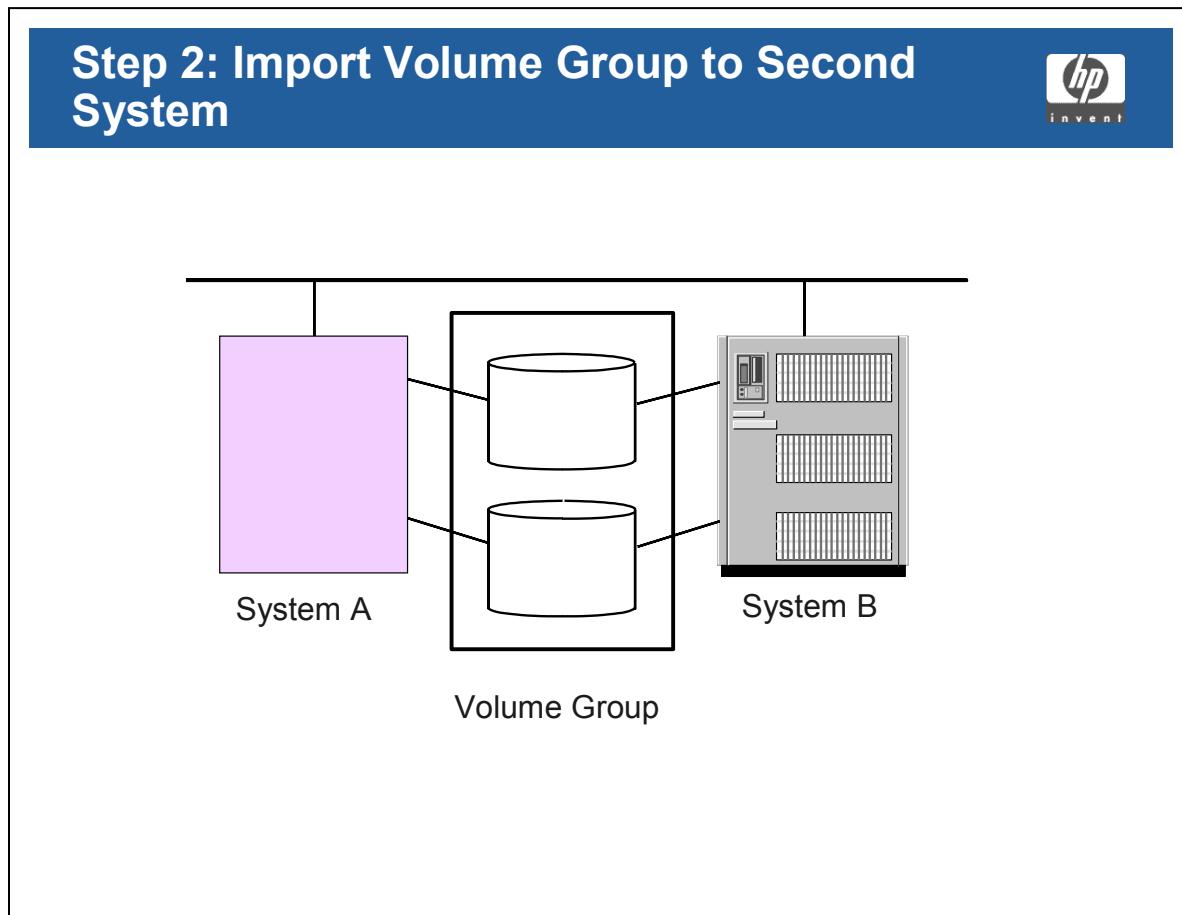
4. Make a mount point directory for the file system.

```
# mkdir /mcsq
```

5. Mount the file system to the mount point directory.

```
# mount -F vxfs /dev/vg01/mcsq /mcsq
```

#### 4-4. SLIDE: Step 2: Import Volume Group to Second System



#### Student Notes

The second step, once the shared volume group has been successfully created, is to *import* the volume group to the second system in the cluster.

#### Export LVM Configuration to the Second System

1. Create a LVM map file for the volume group.

```
[System A] # vgexport -v -p -m /tmp/vg01.mapfile /dev/vg01
```

2. Copy the LVM map file over to the other nodes in the cluster.

```
[System A] # ftp System_B  
ftp> put /tmp/vg01.mapfile
```

3. Create a directory on the other nodes for the volume group.

```
[System B]# mkdir /dev/vg01
```

**Module 4**  
**LVM for Serviceguard**

4. Create the group file on the other nodes for the volume group.

```
[System B]# mknod /dev/vg01/group c 64 0x010000
```

5. Import the volume group onto the other systems.

```
[System B]# vgimport -v -m /tmp/vg01.mapmapfile /dev/vg01 \
/dev/dsk/c_d_t0 /dev/dsk/c_d_t0
```

6. The volume group must be activated to perform the command in step 7.

```
[System B]# vgchange -a y vg01
```

7. Create a configuration backup file

```
[System B]# vgcfgbackup /dev/vg01
```

8. Be sure to deactivate your volume group, as we don't want it active on both nodes simultaneously.

```
[System B]# vgchange -a n vg01
```

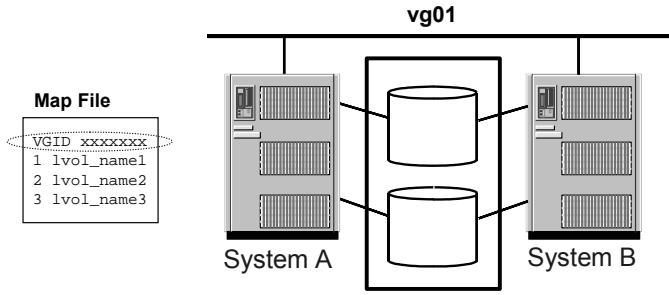
## 4-5. SLIDE: VG Export and Import -s Option

### VG Export and Import -s Option



```
vgexport -v -p -s -m <mapfilename> <vg_name>
create mapfile without removing the VG,
and save the VGID for the vgimport command
```

```
vgimport -v -s -m <mapfilename> <vg_name>
scan for disks that have the same VGID as in the mapfile
```



```
vgexport -v -p -s -m \
/etc/lvmconf/vg01.map vg01
```

```
vgimport -v -s -m \
/etc/lvmconf/vg01.map vg01
```

### Student Notes

The `-s` option was introduced to the `vgexport/vgimport` commands with HP-UX 10.10. This option allows a volume group to be imported without having to specify the disk device file names.

This option is useful when working with large disk arrays, like the XP1024, where determining the disk device file names to use during a `vgimport` can be complex and difficult.

The `vgexport` command with the `-s` option causes the VGID of the volume group being exported to be saved in the map file. Storing the VGID in the map file allows the volume group to be easily imported without specifying disk device file names.

The `vgimport` command with the `-s` option uses the saved VGID in the map file to scan the disks for matches. Physical volumes containing a matching VGID are automatically included during the `vgimport`. This eliminates the need to explicitly specify which physical volumes are being imported.

## 4-6. SLIDE: LVM Issues with Serviceguard

### LVM Issues for Using Serviceguard



#### Some important issues when using LVM within Serviceguard:

- The `/etc/lvmrc` file needs to be modified:  
`AUTO_VG_ACTIVATE=0`  
`custom_vg_activation { }`
- The Volume Group and Logical Volume major and minor numbers MUST be consistent
- The Cluster only needs to be halted to modify the cluster lock configuration and a very few other tasks

## Student Notes

### The `/etc/lvmrc` File

When using volume groups within Serviceguard, the volume groups should *not* be activated automatically at boot time. Serviceguard will activate a package's defined volume group when the package is started on the node where the package is running. Remember, only one node can activate a volume group at a time.

The default behavior of the boot process is to activate all known volume groups at boot time. This needs to be changed for cluster marked volume groups. To modify the default behavior at boot time, edit `/etc/lvmrc` and make the following changes:

- Set `AUTO_VG_ACTIVATE=0`. This will turn off the automatic activation of all volume groups at boot time.

Specify selected volume groups to be activated at boot time in the `custom_vg_activation` routine. All volume groups (except `vg00`) to be activated when the system boots should be listed here. These volume groups do not include cluster marked volume groups.

## **Consistent Major and Minor numbers**

The major and minor numbers used to configure cluster volume groups must be the same across all systems in the cluster.

For example, if system A configures a shared volume group with minor number 0x010000 and system B configures the same volume group with minor number 0x020000, then a cross link error will occur when the cluster comes up.

All major and minor numbers for both volume groups and logical volumes must be consistent across all machines in the cluster.

## **Adding and Modifying LVM Configurations**

Most LVM modifications can be made while the cluster is up and running. The only modifications that require the cluster to be brought down are modifying the configuration of the cluster lock disk and a few other very specific occurrences. All other LVM modifications, even modifications which require a `vgimport` again, can be done while the cluster is up.

When does `vgimport` need to be re-executed? The `vgimport` command needs to be re-executed any time new LVM device files are created. For example, adding a new logical volume or physical volume to an existing shared volume group would require a `vgexport` and `vgimport` on the other systems.

If the modification only changes an LVM characteristic, such as adding three-way mirroring or extending a logical volume, then no `vgimport` is necessary because no new device files were created.

For example, assume `vg01` is exclusively activated on system A, and a new logical volume is created in `vg01`. System B will have no device files for this new logical volume, so system B will have to `vgexport vg01` and then `vgimport vg01` in order to see the new device files for the newly created logical volume. The `vgimport` and `vgexport` are done while `vg01` is exclusively activated on system A. Note that to run `vgimport` and `vgexport`, the volume group never needed to be activated on system B.

## 4–7. LAB: Building a Shared Volume Group

## Directions

The concepts in this lab were discussed in detail in the “Hands-On with Logical Volume Manager” class.

This lab will use these concepts to build a shared volume group that will later be used in a Serviceguard cluster.

Start here by identifying systems to be node1 and node2.

node1: \_\_\_\_\_ (Please write the hostname here)

node2: \_\_\_\_\_ (Please write the hostname here)

## Part 1: Building the Volume Group on Node1

1. On what will hereafter be designated **node1**, list all existing volume groups so that we do not attempt to use an already-assigned disk.
  2. Next, make a directory for a new volume group called **vg01**.
  3. Create a control file for the new volume group.

4. Select two disks not already assigned to any other volume groups. These disks must be sharable between the nodes (i.e. not internal disks). These are the disks we will use for our new volume group.

```
# ioscan -fnC disk
```

When you have selected your disks, write the names of their device files here:

---

first disk, block file name

---

first disk, raw file name

---

second disk, block file name

---

second disk, raw file name

5. Prepare these two disks to be used within the new volume group.

6. Now create the new volume group.

## **Part 2: Setting Up a Logical Volume for Serviceguard**

1. Continuing on `node1`, create a 200MB logical volume named `sglvol1`.
2. Mirror the new `sglvol1` logical volume.
3. Next, create a journal file system in the `sglvol1` logical volume.
4. Create a mount point directory called `/sgdata1`.
5. Mount the new file system to the mount point directory.
6. Finally, for demonstration purposes, copy some files into the file system so that, during a package failover, we can see these files "arrive" on the adoptive node. After copying the files, list the directory contents to see which files were copied.

## Part 3: Importing LVM Configurations to a Second System

1. While still on **Node1**, export the LVM configuration information using "preview" mode.

---

**NOTE:** The following message is just a warning; notice that the mapfile is created anyway.

***Warning: Volume group vgname is still active.***

---

2. Use `ftp` or `rcp` to transfer the map file over to the second system, **Node2**. If you choose `rcp`, you will likely need to add appropriate entries to your `~root/.rhosts` files first.
3. Now, on Node2, create the directory for the volume group (to be imported).
4. Continuing on **Node2**, create the control file for the volume group.

5. Import the volume group.

---

*NOTE:* You can ignore the warning messages about different IDs.

---

Create a mount point directory. This must be the same as on Node1. After creating the mount point directory, activate the volume group in "read-only" mode, mount the file system in "read-only" mode, and view the files previously placed there, thus proving that the volume group / logical volume / file system really is shared. Also, create a backup copy of the LVM configuration. After viewing the files, unmount the file system and deactivate the volume group.

6. Congratulations on successfully building a shared volume group. We will use this volume group when we build our cluster in a later module.

---

## 4-8. LAB Solution: Building a Shared Volume Group

### Directions

The concepts in this lab were discussed in detail in the “Hands-On with Logical Volume Manager” class.

This lab will use these concepts to build a shared volume group that will later be used in a Serviceguard cluster.

Start here by identifying systems to be `node1` and `node2`.

`node1:` \_\_\_\_\_ (Please write the hostname here)

`node2:` \_\_\_\_\_ (Please write the hostname here)

### Part 1: Building the Volume Group on Node1

1. On what will hereafter be designated `node1`, list all existing volume groups so that we do not attempt to use an already-assigned disk.

**Answer:**

```
# strings /etc/lvmtab
```

2. Next, make a directory for a new volume group called `vg01`.

**Answer:**

```
# mkdir /dev/vg01
```

3. Create a control file for the new volume group.

**Answer:**

```
# mknod /dev/vg01/group c 64 0x010000
```

Module 4  
**LVM for Serviceguard**

4. Select two disks not already assigned to any other volume groups. These disks must be sharable between the nodes (i.e. not internal disks). These are the disks we will use for our new volume group.

```
# ioscan -fnC disk
```

When you have selected your disks, write the names of their device files here:

---

first disk, block file name

---

first disk, raw file name

---

second disk, block file name

---

second disk, raw file name

5. Prepare these two disks to be used within the new volume group.

**Answer:**

```
# pvcreate [ -f ] /dev/rdsk/cWtXd0
# pvcreate [ -f ] /dev/rdsk/cYtZd0
```

6. Now create the new volume group.

**Answer:**

```
# vgcreate vg01 /dev/dsk/cWtXd0 /dev/dsk/cYtZd0
```

## Part 2: Setting Up a Logical Volume for Serviceguard

- Continuing on node1, create a 200MB logical volume named `sglvol1`.

**Answer:**

```
# lvcreate -L 200 -n sglvol1 vg01
```

- Mirror the new `sglvol1` logical volume.

**Answer:**

```
# lvextend -m 1 /dev/vg01/sglvol1
```

- Next, create a journaled file system in the `sglvol1` logical volume.

**Answer:**

```
# newfs -F vxfs /dev/vg01/rsglvol1
```

- Create a mount point directory called `/sgdata1`.

**Answer:**

```
# mkdir /sgdata1
```

- Mount the new file system to the mount point directory.

**Answer:**

```
# mount -F vxfs /dev/vg01/sglvol1 /sgdata1
```

- Finally, for demonstration purposes, copy some files into the file system so that, during a package failover, we can see these files "arrive" on the adoptive node. After copying the files, list the directory contents to see which files were copied.

**Answer:**

```
# cp /sbin/lv* /sgdata1/.  
# ll /sgdata1
```

## Part 3: Importing LVM Configurations to a Second System

1. While still on **Node1**, export the LVM configuration information using "preview" mode.

---

**NOTE:** The following message is just a warning; notice that the mapfile is created anyway.

***Warning: Volume group vgname is still active.***

---

**Answer:**

```
# vgexport -v -p -s -m /tmp/vg01.map vg01
```

2. Use **ftp** or **rcp** to transfer the map file over to the second system, **Node2**. If you choose **rcp**, you will likely need to add appropriate entries to your **~root/.rhosts** files first.

**Answer:**

```
# rcp /tmp/vg01.map <Node2>:/tmp/ .
```

3. Now, **on Node2**, create the directory for the volume group (to be imported).

**Answer:**

```
# mkdir /dev/vg01
```

4. Continuing on **Node2**, create the control file for the volume group.

**Answer:**

```
# mknod /dev/vg01/group c 64 0x010000
```

5. Import the volume group.

---

**NOTE:** You can ignore the warning messages about different IDs.

---

**Answer:**

```
# vgimport -v -s -m /tmp/vg01.map vg01
```

6. Create a mount point directory. This must be the same as on Node1. After creating the mount point directory, activate the volume group in "read-only" mode, mount the file system in "read-only" mode, and view the files previously placed there, thus proving that the volume group / logical volume / file system really is shared. Also, create a backup copy of the LVM configuration. After viewing the files, unmount the file system and deactivate the volume group.

## **Answer:**

```
# mkdir /sgdata1
# vgchange -a r vg01
# mount -F vxfs -o ro /dev/vg01/sglvoll /sgdata1
# ll /sgdata1
# vgcfgbackup vg01    ← Create a backup copy of the LVM configuration
# umount /sgdata1
# vgchange -a n vg01
```

- Congratulations on successfully building a shared volume group. We will use this volume group when we build our cluster in a later module.



---

# **Module 5 — Cluster Concepts and Configuration**

## **Objectives**

Upon completion of this module, you will be able to do the following:

- Describe the differences between heartbeat LAN cards, stationary LAN cards, and standby LAN cards.
- Configure a Serviceguard cluster using a cluster lock disk or using Quorum Server technology.
- Configure the heartbeat-polling interval for a Serviceguard cluster.
- Describe three functions performed by the Serviceguard cluster manager daemon.
- Describe the major requirement needed to bring up a Serviceguard cluster.
- Describe Access Control Policies for a cluster
- Implement the systematic procedure for defining, configuring, and bringing up a Serviceguard cluster.

## 5-1. SLIDE: Definition of a Cluster

### Definition of a Cluster



- A Cluster is a collection of nodes which cooperate together to improve the availability of applications running on the nodes.
- Two major benefits of a cluster during downtime events ensure:
  - High Availability when another node takes over. ( Unplanned )
  - The cluster can be used for rolling upgrades. ( Planned )

### Student Notes

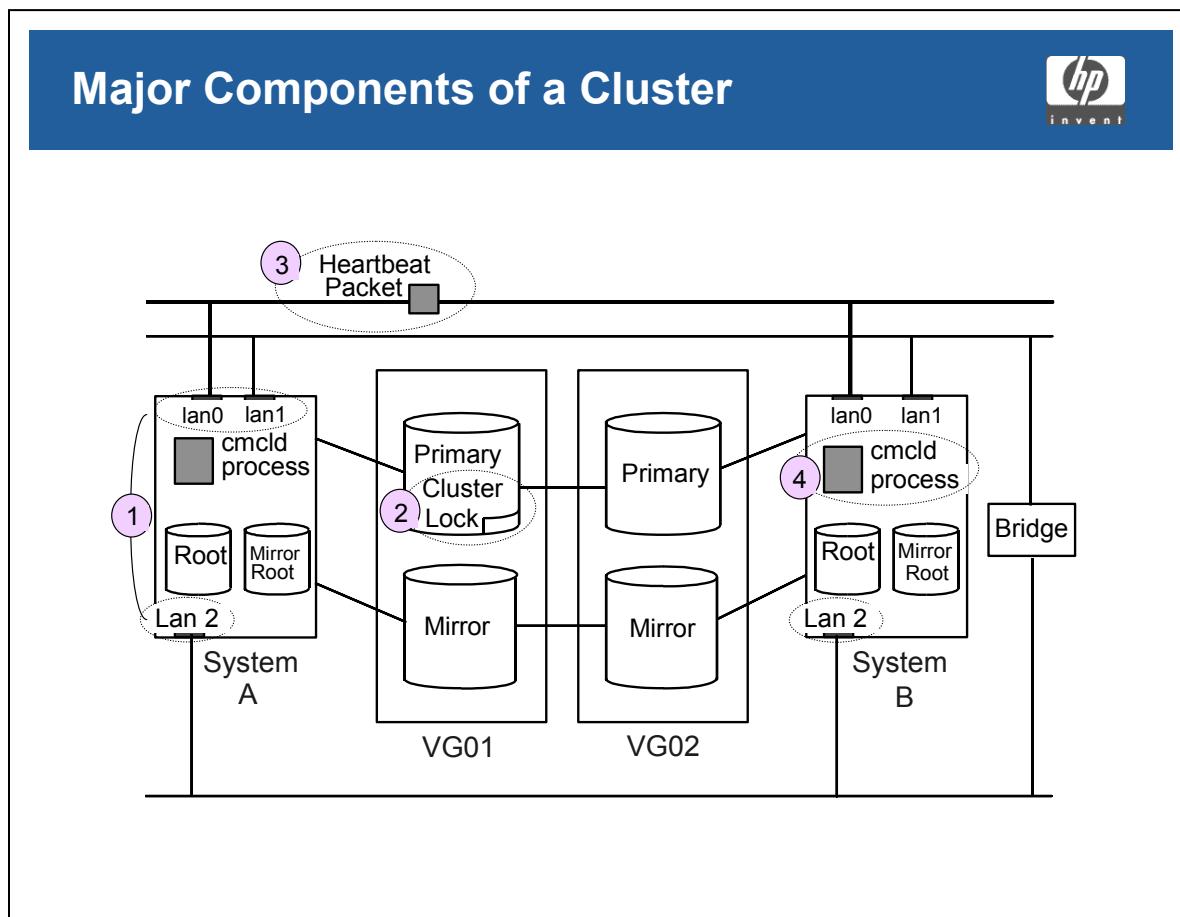
A Serviceguard cluster is a group of networked HP 9000 business servers (nodes) having sufficient redundancy of software and hardware that a single point of failure will not significantly disrupt service. HP 9000 workstations cannot be clustered nodes.

In order to provide a high level of availability, a typical cluster uses redundant system components, like two LANs, two LAN cards, two SCSI cards, and mirrored or RAID disks.

In addition to hardware redundancy, the software must support the capability to transfer applications to another system in the event of a failure. Serviceguard provides the following support:

- In the case of a system failure (CPU, memory, kernel hangs), applications are transferred from the failed system to a functioning system within the cluster in a minimum amount of time.
- In the case of LAN failure, the IP addresses are transferred to a standby LAN card.
- For software failures, the application can be restarted on the same node or switched to another node within the cluster.

## 5–2. SLIDE: Major Components of a Cluster



### Student Notes

The slide shows a sample cluster along with all the major components which make up the cluster. The major components discussed in this module include the following:

1. **LAN interface cards** can be used for data traffic, heartbeat traffic, or for standby purposes.
2. The **cluster lock disk** (shown here), or a Quorum Server (discussed later) is used only during cluster reformation and only for specific situations.
3. **Heartbeat packets** are sent between the nodes to maintain status information.
4. The **cmclid** process is the main Serviceguard process. It is most often referred to as the cluster daemon and it runs on every node within the cluster.
5. Each system in the cluster contains its own root disk. The root disk is mirrored on both systems adding additional high availability.

### 5-3. SLIDE: Network Interface Configuration

## Network Interface Configuration

```
NETWORK_INTERFACE lan0
HEARTBEAT_IP 15.19.83.121
NETWORK_INTERFACE lan1
STATIONARY_IP 192.20.45.121
NETWORK_INTERFACE lan2
# Possible standby network interfaces for lan1 or lan0
NETWORK_POLLING_INTERVAL 2000000
```

Excerpt from Cluster-wide Configuration ASCII file

#### Student Notes

The slide shows the three different ways a LAN card can be configured within the Serviceguard environment.

The bottom of the slide shows what the configurations would look like in the cluster configuration file.

#### HEARTBEAT\_IP

LAN cards configured as HEARTBEAT\_IP are used for heartbeat traffic and potentially for data traffic. A minimum of one HEARTBEAT\_IP must be defined for each node in the cluster.

A card configured with a heartbeat IP address will be used by the cmcld process to communicate with other nodes within the cluster. One such communication is heartbeat messages. These messages are sent by each node in the cluster to the other nodes to indicate they are still alive.

Conversely, each node looks for these heartbeat messages from the other nodes. If heartbeat messages are not received within a prescribed amount of time, the cluster will try to reform minus the node not sending heartbeat messages.

Note that IPv6 heartbeat is not supported, but an IPv6 LAN IS supported within a cluster.

### **STATIONARY\_IP**

LAN cards configured as STATIONARY\_IP are used for data traffic only. Serviceguard monitors these LAN cards; however, no heartbeats or cluster communication packets will travel across the card.

It is recommended that STATIONARY\_IP addresses only be used when two or more heartbeat IP addresses already exist. For redundancy of heartbeat messages, it is better to configure a LAN card as HEARTBEAT\_IP to provide redundancy and to protect against network congestion on other LANs.

The term **stationary** is used to indicate that the IP address will never leave the node. However, the IP address could failover to a standby card within the node.

### **Standby (Indicated by a Blank Definition)**

LAN cards with no defined configuration will automatically be used as standby LAN cards. The term **standby** indicates the card will be used should one of the configured LAN cards (HEARTBEAT\_IP or STATIONARY\_IP) fail.

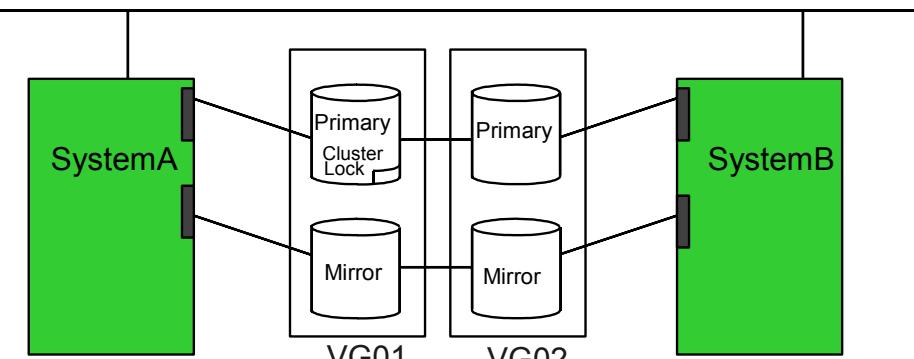
In the event of a failure, all the IP addresses on the failed LAN card will be moved over to the standby LAN card. Standby LAN cards must be on the same physical network as the LAN card they are backing up. This is to allow all other systems to continue communications with the local system after the IP addresses fail over.

In example on the slide, lan0 is being backed up by lan2 on both systems. The standby LAN card cannot be a backup for lan1, because lan2 and lan1 are not on the same physical network.

## 5-4. SLIDE: Cluster Lock Configuration Using LVM Disks

### Cluster Lock Configuration Using LVM Disks





<code>FIRST_CLUSTER_LOCK_VG</code>	/dev/vg01
<code>NODE_NAME</code>	SystemA
<code>FIRST_CLUSTER_LOCK_PV</code>	/dev/dsk/c1t4d0

Excerpt from  
Cluster-wide  
Configuration  
ASCII file

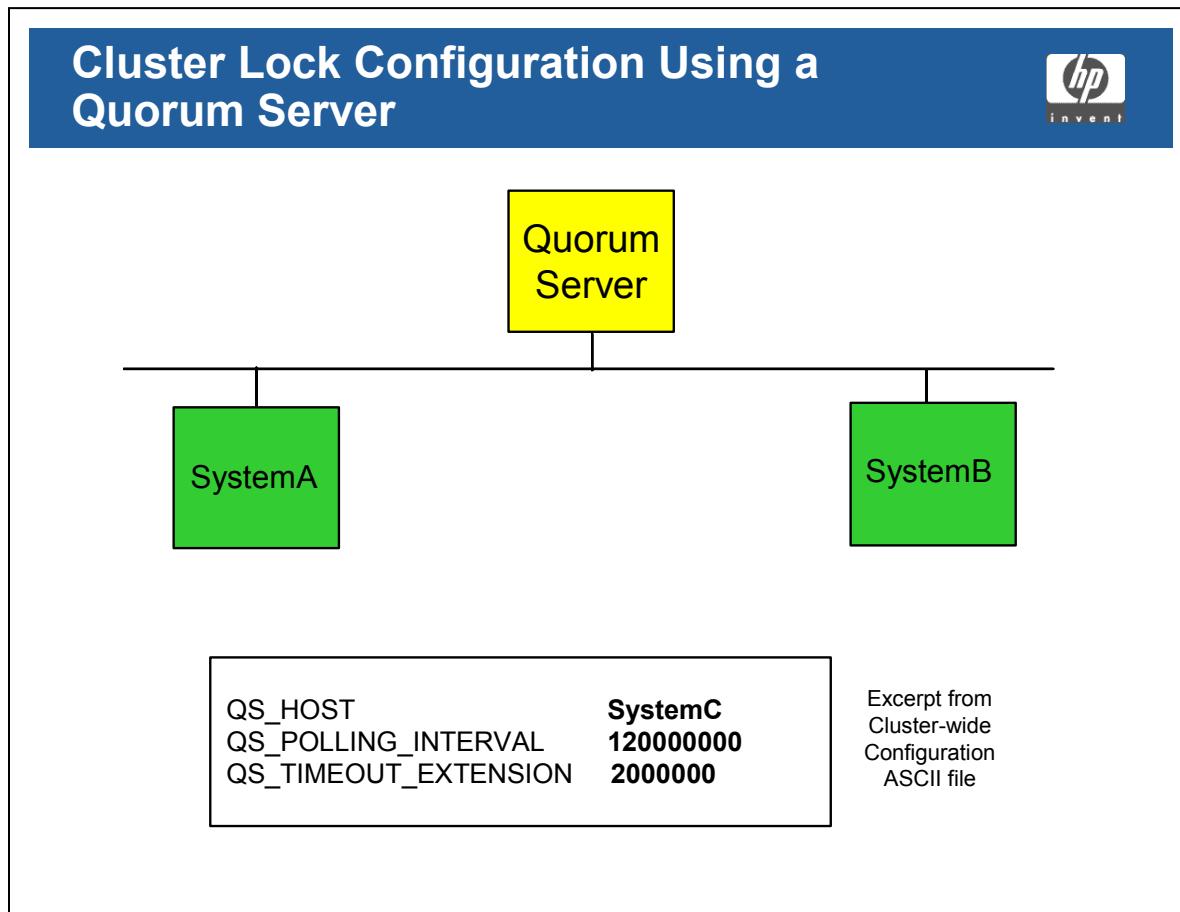
### Student Notes

The cluster lock is a disk located in a volume group shared by all nodes in the cluster. The cluster lock volume group and physical volume are identified in the cluster configuration file.

The cluster lock is used as a tiebreaker only for situations in which the running cluster fails and, in the attempt to reform the cluster, the cluster is split into two subclusters of equal size. Each subcluster will attempt to acquire the cluster lock. The subcluster that gets the cluster lock will form the new cluster. The subcluster that does not get the cluster lock will panic in order to allow existing applications to failover to the new cluster.

In the example on the slide, assume that communications between SystemA and SystemB fails. The failure of the cluster would yield two one-node subclusters. Because each subcluster contains 50 percent of the nodes, both subclusters would try to obtain the cluster lock disk as a means of breaking the tie to determine which subcluster would get to form the new cluster. The subcluster that does not get the cluster lock disk panics.

## 5-5. SLIDE: Cluster Lock Configuration Using a Quorum Server



### Student Notes

An available system, external to the Serviceguard Cluster, could be used to provide Cluster Lock services instead of the traditional LVM disks. This external system is configured as a "Quorum Server" by first using `swinstall` to install the Quorum Server software, and then configuring the system. We will setup a Quorum server in the lab for this module.

Starting with Quorum Server version 2.0, the software will run on both HP-UX and Linux clusters. Additionally, the Quorum Server can be configured as a package in a cluster different than the cluster which is requesting Quorum Server functionality.

Other than the above discussion, all facets of Cluster Lock services are the same as discussed on the previous page.

When configuring your cluster to use a quorum server, the following lines are added to the cluster ascii file:

QS_HOST	quorum_server_hostname
QS_POLLING_INTERVAL	120000000
QS_TIMEOUT_EXTENSION	2000000 # optional

Module 5  
**Cluster Concepts and Configuration**

The QS\_POLLING\_INTERVAL, in microseconds, is the frequency in which the cluster nodes check for TCP connectivity to the Quorum Server

The QS\_TIMEOUT\_EXTENSION relates to the amount of time needed to access the Quorum Server when polling and when a lock is needed. If your environment has added network latency, then it may be necessary to configure this parameter to allow more time to access the Quorum Server before the request ( a TCP connectivity check or a lock request ) times out. By default, there is a Quorum Server timeout that is calculated using the current heartbeat and node timeout values. You would use the QS\_TIMEOUT\_EXTENSION parameter if you need more than the default timeout. Extending this value will increase the cluster reformation time.

### **Configuring a Quorum Server**

1. Choose a system (not within the cluster) to be the Quorum Server.
2. Install Quorum Server software (Product B8467BA) on this server.
3. Create an authorization file, which allows specific hosts to obtain quorum services. This file must be named `/etc/cmcluster/qs_authfile` and will contain the list of all cluster nodes that will access quorum services from the quorum server. Use one line per node.
4. By default, quorum service run-time messages go to `stdout` and `stderr`, so you may want to redirect these messages to a logfile. Create the following directory for this logfile: `/var/adm/qs`
5. Edit the `/etc/inittab` file and add the following line at the bottom:  

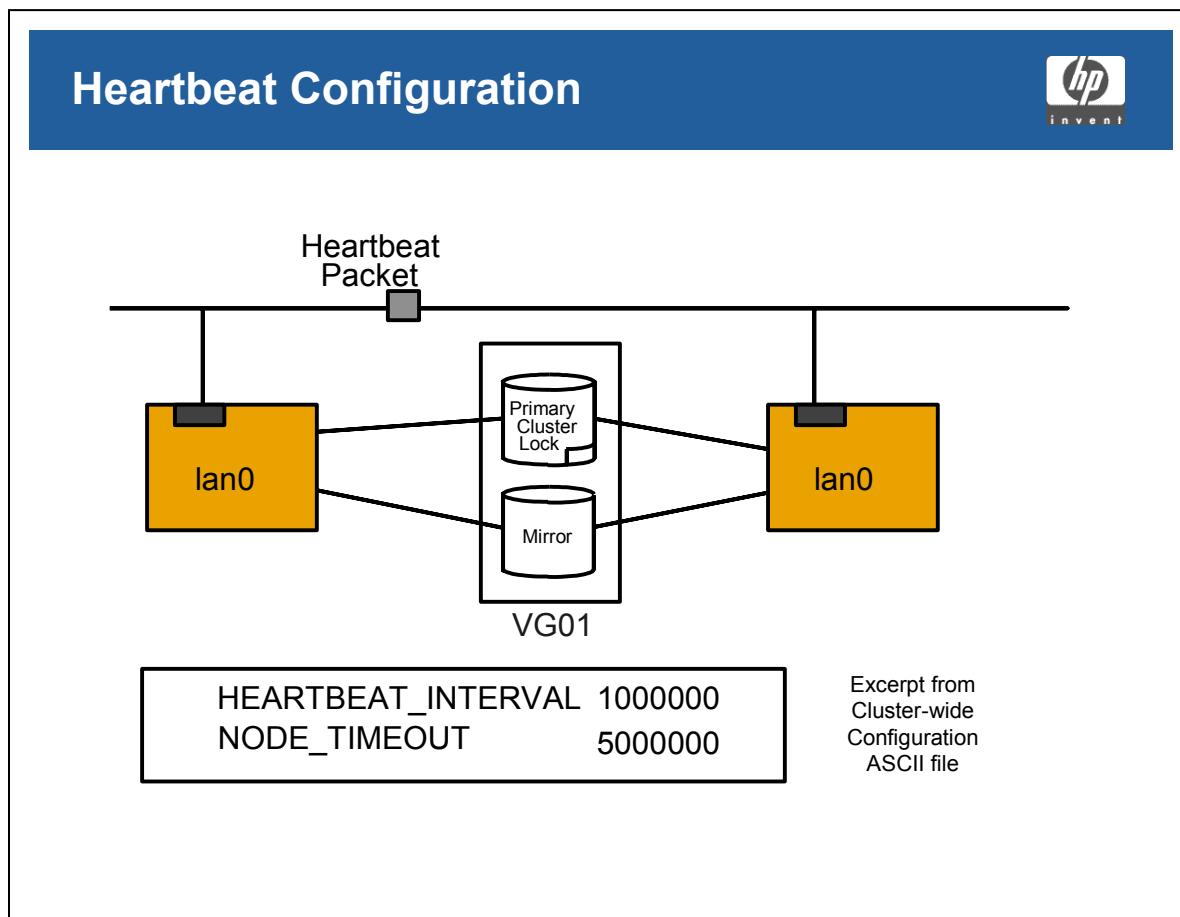
```
qs:345:respawn:/usr/lbin/qs    >> /var/adm/qs/qs.log  2>&1
```
6. Start the Quorum Server as follows: `# init q`

---

**Note:** After starting the quorum server software, if changes are made to `/etc/cmcluster/qs_authfile`, the file will need to be re-read.  
This is done by executing the following command: `# /usr/lbin/qs -update`

More information on the `/usr/lbin/qs` command can be found in the man pages (on the system running the Quorum Server software).

## 5–6. SLIDE: Heartbeat Configuration



### Student Notes

Heartbeat packets are the primary mechanism used by nodes within a cluster to communicate status information.

Each node sends a heartbeat message to the cluster coordinator at every HEARTBEAT\_INTERVAL number of microseconds specified in the cluster configuration file. The cluster coordinator looks for this message from each node, and if it is not received within NODE\_TIMEOUT time, then the cluster is reformed minus the node no longer sending heartbeat messages.

The NODE\_TIMEOUT parameter defaults to 2 seconds. However, for a significant portion of installations, a setting of 5 to 8 seconds is more appropriate. The maximum value recommended for NODE\_TIMEOUT is 30 seconds.

If heartbeat and data are sent over the same LAN subnet, data congestion may cause heartbeat packets not to reach the cluster coordinator. This could cause a cluster reformation even though all nodes in the cluster are healthy. To prevent this situation, it is recommended that multiple HEARTBEAT\_IPs be configured within a cluster.

## 5-7. SLIDE: cmcld Process

### cmcld Process

The cmcld process

- Sends and receives Heartbeats
- Checks the LAN cards for functionality
- Monitors the service processes of packages

## Student Notes

The cluster manager daemon (`cmcld`) is the primary Serviceguard process. This process initializes the cluster, monitors the health of the cluster, and recognizes when a failure within the cluster occurs. This process also performs the reformation of the cluster when a node joins or leaves the cluster.

Every node in the cluster will have a `cmcld` process. During the formation of the cluster, one of the `cmcld` processes will be selected to be the **cluster coordinator**. The cluster coordinator has the additional responsibility to be the central focal point for all heartbeat messages.

## 5–8. SLIDE: Cluster Formation Requirements

### Cluster Formation Requirements



#### Cluster Lock Mechanism

- Required for 2 nodes ( Quorum Server or LVM Disks )
- Recommended for 3 or 4 nodes ( Quorum Server or LVM Disks )
- Optional for 5 or more nodes ( Quorum Server only )
- Should be on separate power sources from other systems in cluster

#### Node Attendance

- By default, 100% node attendance is required for initial cluster formation

#### Network Requirements

- All nodes in the cluster must be on same subnet
- Configured LAN cards in the same system MUST be on separate subnets
- At least one HEARTBEAT\_IP network is configured

### Student Notes

#### Cluster Lock Disk

A cluster lock disk is required for the formation of a two-node cluster. Without a cluster lock disk, the cluster would not be able to reform should one node fail, causing the entire cluster to go down. Cluster lock functionality may be implemented by using either LVM disks or using a Quorum server.

With three or four nodes, a cluster lock is recommended but not required. For three nodes, there would never be a situation where the nodes are divided 50/50. However, if a node is pulled out of the cluster for maintenance purposes, then the cluster lock would be used should one of the remaining two nodes in the cluster fail.

For four nodes, the cluster lock disk would be used when the nodes are separated into two, 2-node sub clusters. The lock is not required, because the chance of being split into two, equally sized halves is very small for most environments.

For five or more nodes, a cluster lock using LVM disks is not supported. While a Quorum Server could be used, it would be optional.

## **100 Percent Node Attendance**

To initially bring up any cluster of any size, 100 percent node attendance is required. Without this requirement, the same cluster could potentially form twice with two different sets of nodes.

When a request is made by one of the nodes to form/bring up the cluster, all nodes in the cluster are notified of the request. If one of the nodes already has the cluster up, the other nodes will join the already existing cluster. If all nodes return confirmation of the request and no nodes have already formed a cluster, a new cluster formation will be performed. If however, all nodes do not return confirmation of the request, then the formation of the cluster will wait for the outstanding confirmations to arrive. The amount of time that the nodes wait for outstanding confirmations is AUTO\_START\_TIMEOUT, which is defined in the cluster configuration file.

## **Network Requirements**

In order to form a new cluster, every node must have at least one HEARTBEAT\_IP address configured. These HEARTBEAT\_IP addresses must be on the same subnet (if a node has two HEARTBEAT\_IP addresses, then only one has to be on the common subnet).

If a node has multiple LAN cards configured, the LAN cards must all be on different subnets. Routing protocols get confused when two LAN cards in the same system are configured for the same subnet.

## 5–9. SLIDE: Steps to Configure a Cluster

### Steps to Configure a Cluster



#### 1. Build cluster configuration file

```
cmquerycl -v -C cmclconfig.ascii -n node1 -n node2...
```

#### 2. Compile and distribute binary file

```
cmcheckconf -v -C cmclconfig.ascii  
cmapplyconf -v -C cmclconfig.ascii
```

#### 3. Start the Serviceguard daemons

```
cmrunc1 -v
```

#### 4. If necessary, halt the cluster

```
cmhaltcl -v
```

## Student Notes

To configure a Serviceguard cluster, the three basic steps are:

1. **Build the cluster configuration ASCII file.** The file that the user creates defines the nodes, the disks, the LAN cards, and any other resources that are to be part of the cluster. The command used to help build the cluster configuration ASCII file is called `cmquerycl`. If using a Quorum Server to provide cluster lock services, be sure to have configured the QS software on another host. Then, in the `cmquerycl` command, include the `-q` option, specifying the `QS_HOST`.
2. **Compile and distribute the cluster binary file.** Once the cluster configuration ASCII file is created, it needs to be compiled into a binary format that the Serviceguard daemons can access efficiently. This compiled file is then distributed automatically to all other nodes in the cluster. The command used to compile and distribute the binary configuration file is called `cmapplyconf`.
3. **Start the Serviceguard daemons.** Once the binary file is distributed, the Serviceguard daemon, `cmcl`, can be started. This daemon *cannot* be started directly, and should be started only through the `cmbunc1` command.

## 5-10. SLIDE: Step 1: Build Cluster Configuration ASCII File

### Step 1: Build Cluster Configuration ASCII File

Use `cmquerycl` command to build cluster configuration ASCII file

cmqlconfd process

cmqlconfd process

cmqlconfd process

- CLUSTER\_NAME
- FIRST\_CLUSTER\_LOCK\_VG
- NODE\_NAME
  - NETWORK\_INTERFACE
  - HEARTBEAT\_IP or STATIONARY\_IP or blank
  - FIRST\_CLUSTER\_LOCK\_PV
  - SERIAL\_DEVICE\_FILE
- HEARTBEAT\_INTERVAL
- NODE\_TIMEOUT
- AUTOSTART\_TIMEOUT
- NETWORK\_POLLING\_INTERVAL
- MAX\_CONFIGURED\_PACKAGES
- VOLUME\_GROUP

**Steps to Configure a Cluster**

1. Build Cluster Configuration File
2. Compile and Distribute Binary File
3. Start Serviceguard Daemons

### Student Notes

The first step in building the cluster is to create a “cluster-wide configuration ASCII file” which describes the overall contents of the cluster. The cluster-wide configuration ASCII file should be created in the `/etc/cmcluster` directory.

The `cmquerycl` command is used to gather the node-specific information and to create a template. The syntax for the `cmquerycl` command is:

```
# cmquerycl -v -C cmclconfig.ascii -n node1 -n node2
```

Once the template is created, it can easily be edited and customized with any editor. At a minimum, the `CLUSTER_NAME` and `NODE_TIMEOUT` variables should be modified in the template. Other values should be modified as appropriate.

The fields in the cluster configuration template ASCII file are described below:

<i>CLUSTER_NAME</i>	The name of the cluster as it will appear in the output of <code>cmviewcl</code> and other commands. This name must be unique. The cluster name can contain up to 40 characters.
<i>FIRST_CLUSTER_LOCK_VG</i>	The volume group from which the cluster lock disk is selected.
<i>NODE_NAME</i>	The hostname of a system that will be a node in the cluster. This field must be defined for each system in the cluster. The node name can contain up to 31 characters.
<i>NETWORK_INTERFACE</i>	The network interface as listed by <code>lanscan</code> (e.g. <code>lan1</code> ).
<i>HEARTBEAT_IP</i>	The IP address of the LAN interface, which will carry heartbeat messages. A minimum of one <i>HEARTBEAT_IP</i> must be defined.
<i>FIRST_CLUSTER_LOCK_PV</i>	The name of the physical volume within the <i>FIRST_CLUSTER_LOCK_VG</i> that will have the cluster lock written on it. The physical volume group identifier can contain up to 40 characters.
<i>SERIAL_DEVICE_FILE</i>	The name of the device file that corresponds to the RS232 serial port that will carry the serial heartbeat that will be used only in the case where the heartbeat communications is interrupted. The device name can contain up to 40 characters.
<i>HEARTBEAT_INTERVAL</i>	The interval between transmissions of heartbeat messages from one node to the cluster coordinator. The default value is 1000000 microseconds (one second).
<i>NODE_TIMEOUT</i>	The time (in microseconds) after which a node may decide another node has become unavailable and thereby initiate a cluster reformation. The default value is 2000000 microseconds (2 seconds). A good rule of thumb is to have 2-3 heartbeats within 1 <i>NODE_TIMEOUT</i> .
<i>AUTOSTART_TIMEOUT</i>	The amount of time a node waits for other nodes to become available when bringing up the cluster. All nodes must be present in order for the cluster to come up. The default value is 600000000 microseconds (10 minutes)

***NETWORK\_POLLING\_INTERVAL***

The frequency at which the cluster-identified LAN cards are checked by the cluster manager daemon (cmcl). The default is 2000000 microseconds (2 seconds).

***NETWORK\_FAILURE\_DETECTION***

Determines how LAN card failures are handled. If set to INONLY\_OR\_INOUT, the LAN card will be considered down when the inbound message count stops increasing –or– when both inbound & outbound message counts stop increasing. If set to INOUT, both inbound and outbound message counts must stop increasing for the LAN card to be considered down. INOUT is the default.

***MAX\_CONFIGURED\_PACKAGES***

The maximum number of packages that can be configured in the cluster. The default is 150, as is the absolute maximum.

*Access Control Policy*

By default, Serviceguard configuration can only be performed by the system administrator. Access control policies allow the root user to delegate certain tasks to non-root users. The cluster ascii file has entries for USER\_NAME, USER\_HOST, and USER\_ROLE. USER\_NAME is the user to whom certain access will be granted. USER\_HOST is where USER\_NAME can issue Serviceguard commands (not necessarily a node in the cluster). USER\_ROLE defines USER\_NAME's privileges (FULL\_ADMIN, PACKAGE\_ADMIN, or MONITOR). Note: None of the users granted access control are allowed to configure the cluster; this activity is limited only to the root user.

***VOLUME\_GROUP***

The volume groups which the Serviceguard cluster uses. The volume group name can contain up to 39 characters.

## 5-11. SLIDE: Cluster Management Options

### Cluster Management Options



#### Role-Based Access Control

- By default, only the root user can access/control the cluster
- Other users (optionally from other hosts) can be allowed certain privileges

#### Access Control Policy Parameters

- **USER\_NAME** ANY\_USER –or- a list of specific users
- **USER\_HOST** Where the user can execute SG commands
- **USER\_ROLE** MONITOR, PACKAGE\_ADMIN, or FULL\_ADMIN

#### Limitations

- Only the root user can configure a cluster or packages

### Student Notes

Beginning with Serviceguard A.11.16, non-root users can manage/monitor a cluster and/or the packages within it. In the cluster ascii file, there are several options for Access Control. Specifically, `USER_NAME`, `USER_HOST`, and `USER_ROLE`.

For each of these, the options follow:

The first line must be `USER_NAME`, second `USER_HOST`, and third `USER_ROLE`.

1. `USER_NAME` can either be `ANY_USER`, or a maximum of 8 login names from the `/etc/passwd` file on user host.
2. `USER_HOST` is where the user can issue Serviceguard commands. If using Serviceguard Manager, it is the COM (Cluster Object Manager) server. Choose one of these three values: `ANY_SERVICEGUARD_NODE`, or (any) `CLUSTER_MEMBER_NODE`, or a specific node. For node, use the official hostname from domain name server, and not an IP address or fully qualified name.

3. USER\_ROLE must be one of these three values:

- MONITOR: read-only capabilities for the cluster and packages
- PACKAGE\_ADMIN: MONITOR, plus administrative commands for packages in the cluster
- FULL\_ADMIN: MONITOR and PACKAGE\_ADMIN plus the administrative commands for the cluster.

Access control policy does not set a role for configuration capability. To configure a cluster or a package, a user must log on to one of the cluster's nodes as root (UID=0). Access control policy cannot limit root users' access.

MONITOR and FULL\_ADMIN can only be set in the cluster configuration file, and they apply to the entire cluster. PACKAGE\_ADMIN can be set in the cluster or a package configuration file. If set in the cluster configuration file, PACKAGE\_ADMIN applies to all configured packages. If set in a package configuration file, which will be discussed in a later module, PACKAGE\_ADMIN applies to that package only.

Conflicting or redundant policies will cause an error while applying the configuration, and will stop the process.

Example: to configure a role for user oracle from node rx26u891 to administer a cluster and all its packages, enter:

```
USER_NAME    oracle
USER_HOST    rx26u891
USER_ROLE    FULL_ADMIN
```

Note: Access Policies are not supported in a mixed revision cluster. If updating from an earlier version of Serviceguard, and if /etc/cmcluster/clcmnodelist was used to give non-root users access, the update process will convert old cmclnodelist entries into Access Control Policies when every nodes in the cluster is finished updating. Additionally, all are given the role of MONITOR in the Cluster ASCII file.

## 5-12. TEXT PAGE: The cmquerycl command

### Special Note

Here is the actual screen output resulting from the cmquerycl command.  
This output was obtained while using version 11.17 of Serviceguard.

```
root@rx26u230  [/etc/cmcluster]
# cmquerycl -v -C cmclconfig.ascii -n rx26u230 -n rx26u231
Looking for other clusters ... Done
Gathering storage information
Found 79 devices on node rx26u230
Found 79 devices on node rx26u231
Analysis of 158 devices should take approximately 9 seconds
0%----10%----20%----30%----40%----50%----60%----70%----80%----90%----100%
Found 2 volume groups on node rx26u230
Found 2 volume groups on node rx26u231
Analysis of 4 volume groups should take approximately 1 seconds
0%----10%----20%----30%----40%----50%----60%----70%----80%----90%----100%
Note: Disks were discovered which are not in use by either LVM or VxVM.
      Use pvcreate(1M) to initialize a disk for LVM or,
      use vxdiskadm(1M) to initialize a disk for VxVM.
Gathering network information
Beginning network probing
Completed network probing
Network interface lan0 on node rx26u230 couldn't talk to itself.
Network interface lan0 on node rx26u231 couldn't talk to itself.

Node Names:      rx26u230
                  rx26u231

Bridged networks (local node information only - full probing was not
performed):

1      lan1          (rx26u230)
      lan2          (rx26u230)
      lan3          (rx26u230)

2      lan1          (rx26u231)
      lan2          (rx26u231)
      lan3          (rx26u231)

IP subnets:

IPv4:

10.10.0.0      lan1          (rx26u230)
                lan1          (rx26u231)

IPv6:

Possible Heartbeat IPs:
```

Module 5  
**Cluster Concepts and Configuration**

10.10.0.0	10.10.48.130	(rx26u230)
	10.10.48.131	(rx26u231)

Possible Cluster Lock Devices:

/dev/dsk/c20t0d7	/dev/vg01	66 seconds
------------------	-----------	------------

LVM volume groups:

/dev/vg00	rx26u230
/dev/vg01	rx26u230
	rx26u231
/dev/vg00	rx26u231

LVM physical volumes:

/dev/vg00		
/dev/dsk/c2t1d0s2	0/1/1/0.1.0	rx26u230
/dev/vg01		
/dev/dsk/c20t0d7	255/255/0/0.7	rx26u230
/dev/dsk/c20t0d7	255/255/0/0.7	rx26u231
/dev/vg00		
/dev/dsk/c2t1d0s2	0/1/1/0.1.0	rx26u231

LVM logical volumes:

Volume groups on rx26u230:

/dev/vg00/lvol1	FS MOUNTED	/stand
/dev/vg00/lvol2		
/dev/vg00/lvol3	FS MOUNTED	/
/dev/vg00/lvol4	FS MOUNTED	/home
/dev/vg00/lvol5	FS MOUNTED	/oral
/dev/vg00/lvol6	FS MOUNTED	/tmp
/dev/vg00/lvol7	FS MOUNTED	/opt
/dev/vg00/lvol8	FS MOUNTED	/usr
/dev/vg00/lvol9	FS MOUNTED	/var

Volume groups on rx26u231:

/dev/vg00/lvol1	FS MOUNTED	/stand
/dev/vg00/lvol2		
/dev/vg00/lvol3	FS MOUNTED	/
/dev/vg00/lvol4	FS MOUNTED	/home
/dev/vg00/lvol5	FS MOUNTED	/oral
/dev/vg00/lvol6	FS MOUNTED	/tmp
/dev/vg00/lvol7	FS MOUNTED	/opt
/dev/vg00/lvol8	FS MOUNTED	/usr
/dev/vg00/lvol9	FS MOUNTED	/var

Writing cluster data to cmclconfig.ascii.

root@rx26u230 [/etc/cmcluster]

#

## 5-13. TEXT PAGE: Sample Cluster Configuration ASCII File Using LVM Disks for Cluster Lock Services

The following is the cluster-wide ASCII configuration file using Serviceguard version A.11.17, without a Quorum Server ( i.e., using LVM disks for cluster lock functionality )

```
# ****
# ***** HIGH AVAILABILITY CLUSTER CONFIGURATION FILE ****
# ***** For complete details about cluster parameters and how to ****
# ***** set them, consult the Serviceguard manual. ****
# *****

# Enter a name for this cluster. This name will be used to identify the
# cluster when viewing or manipulating it.

CLUSTER_NAME          cluster1

# Cluster Lock Parameters
# The cluster lock is used as a tie-breaker for situations
# in which a running cluster fails, and then two equal-sized
# sub-clusters are both trying to form a new cluster. The
# cluster lock may be configured using only one of the
# following alternatives on a cluster:
#         the LVM lock disk
#         the quorum server
#
#
# Consider the following when configuring a cluster.
# For a two-node cluster, you must use a cluster lock. For
# a cluster of three or four nodes, a cluster lock is strongly
# recommended. For a cluster of more than four nodes, a
# cluster lock is recommended. If you decide to configure
# a lock for a cluster of more than four nodes, it must be
# a quorum server.

# Lock Disk Parameters. Use the FIRST_CLUSTER_LOCK_VG and
# FIRST_CLUSTER_LOCK_PV parameters to define a lock disk.
# The FIRST_CLUSTER_LOCK_VG is the LVM volume group that
# holds the cluster lock. This volume group should not be
# used by any other cluster as a cluster lock device.

# Quorum Server Parameters. Use the QS_HOST, QS_POLLING_INTERVAL,
# and QS_TIMEOUT_EXTENSION parameters to define a quorum server.
# The QS_HOST is the host name or IP address of the system
# that is running the quorum server process. The
# QS_POLLING_INTERVAL (microseconds) is the interval at which
# Serviceguard checks to make sure the quorum server is running.
# The optional QS_TIMEOUT_EXTENSION (microseconds) is used to increase
# the time interval after which the quorum server is marked DOWN.
#
#
# The default quorum server timeout is calculated from the
# Serviceguard cluster parameters, including NODE_TIMEOUT and
# HEARTBEAT_INTERVAL. If you are experiencing quorum server
# timeouts, you can adjust these parameters, or you can include
# the QS_TIMEOUT_EXTENSION parameter.
```

Module 5  
**Cluster Concepts and Configuration**

```
#  
# The value of QS_TIMEOUT_EXTENSION will directly effect the amount  
# of time it takes for cluster reformation in the event of failure.  
# For example, if QS_TIMEOUT_EXTENSION is set to 10 seconds, the cluster  
# reformation will take 10 seconds longer than if the QS_TIMEOUT_EXTENSION  
# was set to 0. This delay applies even if there is no delay in  
# contacting the Quorum Server. The recommended value for  
# QS_TIMEOUT_EXTENSION is 0, which is used as the default  
# and the maximum supported value is 30000000 (5 minutes).  
#  
# For example, to configure a quorum server running on node  
# "qshost" with 120 seconds for the QS_POLLING_INTERVAL and to  
# add 2 seconds to the system assigned value for the quorum server  
# timeout, enter:  
#  
# QS_HOST qshost  
# QS_POLLING_INTERVAL 120000000  
# QS_TIMEOUT_EXTENSION 2000000  
  
FIRST_CLUSTER_LOCK_VG          /dev/vg01  
  
# Definition of nodes in the cluster.  
# Repeat node definitions as necessary for additional nodes.  
# NODE_NAME is the specified nodename in the cluster.  
# It must match the hostname and both cannot contain full domain name.  
# Each NETWORK_INTERFACE, if configured with IPv4 address,  
# must have ONLY one IPv4 address entry with it which could  
# be either HEARTBEAT_IP or STATIONARY_IP.  
# Each NETWORK_INTERFACE, if configured with IPv6 address(es)  
# can have multiple IPv6 address entries(up to a maximum of 2,  
# only one IPv6 address entry belonging to site-local scope  
# and only one belonging to global scope) which must be all  
# STATIONARY_IP. They cannot be HEARTBEAT_IP.  
  
NODE_NAME          rx26u230  
NETWORK_INTERFACE lan1  
HEARTBEAT_IP      10.10.48.130  
NETWORK_INTERFACE lan2  
NETWORK_INTERFACE lan3  
FIRST_CLUSTER_LOCK_PV /dev/dsk/c20t0d7  
# List of serial device file names  
# For example:  
# SERIAL_DEVICE_FILE    /dev/tty0p0  
  
# Possible standby Network Interfaces for lan1: lan2,lan3.  
  
NODE_NAME          rx26u231  
NETWORK_INTERFACE lan1  
HEARTBEAT_IP      10.10.48.131  
NETWORK_INTERFACE lan2  
NETWORK_INTERFACE lan3  
FIRST_CLUSTER_LOCK_PV /dev/dsk/c20t0d7  
# List of serial device file names  
# For example:  
# SERIAL_DEVICE_FILE    /dev/tty0p0  
  
# Possible standby Network Interfaces for lan1: lan2,lan3.  
  
# Cluster Timing Parameters (microseconds).
```

```

# The NODE_TIMEOUT parameter defaults to 2000000 (2 seconds).
# This default setting yields the fastest cluster reformations.
# However, the use of the default value increases the potential
# for spurious reformations due to momentary system hangs or
# network load spikes.
# For a significant portion of installations, a setting of
# 5000000 to 8000000 (5 to 8 seconds) is more appropriate.
# The maximum value recommended for NODE_TIMEOUT is 30000000
# (30 seconds).

HEARTBEAT_INTERVAL          1000000
NODE_TIMEOUT                2000000

# Configuration/Reconfiguration Timing Parameters (microseconds).

AUTO_START_TIMEOUT          6000000000
NETWORK_POLLING_INTERVAL    2000000

# Network Monitor Configuration Parameters.
# The NETWORK_FAILURE_DETECTION parameter determines how LAN card failures are
detected.
# If set to INONLY_OR_INOUT, a LAN card will be considered down when its inbound
# message count stops increasing or when both inbound and outbound
# message counts stop increasing.
# If set to INOUT, both the inbound and outbound message counts must
# stop increasing before the card is considered down.
NETWORK_FAILURE_DETECTION    INOUT

# Package Configuration Parameters.
# Enter the maximum number of packages which will be configured in the cluster.
# You can not add packages beyond this limit.
# This parameter is required.
MAX_CONFIGURED_PACKAGES      150

# Access Control Policy Parameters.
#
# Three entries set the access control policy for the cluster:
# First line must be USER_NAME, second USER_HOST, and third USER_ROLE.
# Enter a value after each.
#
# 1. USER_NAME can either be ANY_USER, or a maximum of
#    8 login names from the /etc/passwd file on user host.
# 2. USER_HOST is where the user can issue Serviceguard commands.
#    If using Serviceguard Manager, it is the COM server.
#    Choose one of these three values: ANY_SERVICEGUARD_NODE, or
#    (any) CLUSTER_MEMBER_NODE, or a specific node. For node,
#    use the official hostname from domain name server, and not
#    an IP addresses or fully qualified name.
# 3. USER_ROLE must be one of these three values:
#    * MONITOR: read-only capabilities for the cluster and packages
#    * PACKAGE_ADMIN: MONITOR, plus administrative commands for packages
#      in the cluster
#    * FULL_ADMIN: MONITOR and PACKAGE_ADMIN plus the administrative
#      commands for the cluster.
#
# Access control policy does not set a role for configuration
# capability. To configure, a user must log on to one of the
# cluster's nodes as root (UID=0). Access control
# policy cannot limit root users' access.
#
# MONITOR and FULL_ADMIN can only be set in the cluster configuration file,

```

## Module 5

### Cluster Concepts and Configuration

```
# and they apply to the entire cluster. PACKAGE_ADMIN can be set in the
# cluster or a package configuration file. If set in the cluster
# configuration file, PACKAGE_ADMIN applies to all configured packages.
# If set in a package configuration file, PACKAGE_ADMIN applies to that
# package only.
#
# Conflicting or redundant policies will cause an error while applying
# the configuration, and stop the process. The maximum number of access
# policies that can be configured in the cluster is 200.
#
# Example: to configure a role for user john from node noir to
# administer a cluster and all its packages, enter:
# USER_NAME    john
# USER_HOST    noir
# USER_ROLE    FULL_ADMIN

# List of cluster aware LVM Volume Groups. These volume groups will
# be used by package applications via the vgchange -a e command.
# Neither CVM or VxVM Disk Groups should be used here.
# For example:
# VOLUME_GROUP      /dev/vgdatabase
# VOLUME_GROUP      /dev/vg02

VOLUME_GROUP      /dev/vg01
```

---

## 5-14. TEXT PAGE: Sample Cluster Configuration ASCII File Using Quorum Server Technology for Cluster Lock Services

The following is the cluster-wide ASCII configuration file using Serviceguard version A.11.17, with a Quorum Server (i.e. an external system ) for cluster lock functionality

```
# ****
# ***** HIGH AVAILABILITY CLUSTER CONFIGURATION FILE ****
# ***** For complete details about cluster parameters and how to ****
# ***** set them, consult the Serviceguard manual. ****
# *****

# Enter a name for this cluster. This name will be used to identify the
# cluster when viewing or manipulating it.

CLUSTER_NAME          cluster1

# Cluster Lock Parameters
# The cluster lock is used as a tie-breaker for situations
# in which a running cluster fails, and then two equal-sized
# sub-clusters are both trying to form a new cluster. The
# cluster lock may be configured using only one of the
# following alternatives on a cluster:
#     the LVM lock disk
#     the quorum server
#
#
# Consider the following when configuring a cluster.
# For a two-node cluster, you must use a cluster lock. For
# a cluster of three or four nodes, a cluster lock is strongly
# recommended. For a cluster of more than four nodes, a
# cluster lock is recommended. If you decide to configure
# a lock for a cluster of more than four nodes, it must be
# a quorum server.

# Lock Disk Parameters. Use the FIRST_CLUSTER_LOCK_VG and
# FIRST_CLUSTER_LOCK_PV parameters to define a lock disk.
# The FIRST_CLUSTER_LOCK_VG is the LVM volume group that
# holds the cluster lock. This volume group should not be
# used by any other cluster as a cluster lock device.

# Quorum Server Parameters. Use the QS_HOST, QS_POLLING_INTERVAL,
# and QS_TIMEOUT_EXTENSION parameters to define a quorum server.
# The QS_HOST is the host name or IP address of the system
# that is running the quorum server process. The
# QS_POLLING_INTERVAL (microseconds) is the interval at which
# Serviceguard checks to make sure the quorum server is running.
# The optional QS_TIMEOUT_EXTENSION (microseconds) is used to increase
# the time interval after which the quorum server is marked DOWN.
#
# The default quorum server timeout is calculated from the
# Serviceguard cluster parameters, including NODE_TIMEOUT and
# HEARTBEAT_INTERVAL. If you are experiencing quorum server
# timeouts, you can adjust these parameters, or you can include
```

## Module 5

### Cluster Concepts and Configuration

```
# the QS_TIMEOUT_EXTENSION parameter.  
#  
# The value of QS_TIMEOUT_EXTENSION will directly effect the amount  
# of time it takes for cluster reformation in the event of failure.  
# For example, if QS_TIMEOUT_EXTENSION is set to 10 seconds, the cluster  
# reformation will take 10 seconds longer than if the QS_TIMEOUT_EXTENSION  
# was set to 0. This delay applies even if there is no delay in  
# contacting the Quorum Server. The recommended value for  
# QS_TIMEOUT_EXTENSION is 0, which is used as the default  
# and the maximum supported value is 30000000 (5 minutes).  
#  
# For example, to configure a quorum server running on node  
# "qshost" with 120 seconds for the QS_POLLING_INTERVAL and to  
# add 2 seconds to the system assigned value for the quorum server  
# timeout, enter:  
#  
# QS_HOST qshost  
# QS_POLLING_INTERVAL 120000000  
# QS_TIMEOUT_EXTENSION 2000000  
  
QS_HOST          10.51.206.148  
QS_POLLING_INTERVAL 300000000  
  
# Definition of nodes in the cluster.  
# Repeat node definitions as necessary for additional nodes.  
# NODE_NAME is the specified nodename in the cluster.  
# It must match the hostname and both cannot contain full domain name.  
# Each NETWORK_INTERFACE, if configured with IPv4 address,  
# must have ONLY one IPv4 address entry with it which could  
# be either HEARTBEAT_IP or STATIONARY_IP.  
# Each NETWORK_INTERFACE, if configured with IPv6 address(es)  
# can have multiple IPv6 address entries(up to a maximum of 2,  
# only one IPv6 address entry belonging to site-local scope  
# and only one belonging to global scope) which must be all  
# STATIONARY_IP. They cannot be HEARTBEAT_IP.  
  
NODE_NAME          rx26u230  
NETWORK_INTERFACE  lan1  
HEARTBEAT_IP      10.10.48.130  
NETWORK_INTERFACE  lan2  
NETWORK_INTERFACE  lan3  
# List of serial device file names  
# For example:  
# SERIAL_DEVICE_FILE /dev/tty0p0  
  
# Possible standby Network Interfaces for lan1: lan2,lan3.  
  
NODE_NAME          rx26u231  
NETWORK_INTERFACE  lan1  
HEARTBEAT_IP      10.10.48.131  
NETWORK_INTERFACE  lan2  
NETWORK_INTERFACE  lan3  
# List of serial device file names  
# For example:  
# SERIAL_DEVICE_FILE /dev/tty0p0  
  
# Possible standby Network Interfaces for lan1: lan2,lan3.  
  
# Cluster Timing Parameters (microseconds).  
# The NODE_TIMEOUT parameter defaults to 2000000 (2 seconds).
```

```

# This default setting yields the fastest cluster reformations.
# However, the use of the default value increases the potential
# for spurious reformations due to momentary system hangs or
# network load spikes.
# For a significant portion of installations, a setting of
# 5000000 to 8000000 (5 to 8 seconds) is more appropriate.
# The maximum value recommended for NODE_TIMEOUT is 30000000
# (30 seconds).

HEARTBEAT_INTERVAL          1000000
NODE_TIMEOUT                2000000

# Configuration/Reconfiguration Timing Parameters (microseconds).

AUTO_START_TIMEOUT          600000000
NETWORK_POLLING_INTERVAL    2000000

# Network Monitor Configuration Parameters.
# The NETWORK_FAILURE_DETECTION parameter determines how LAN card failures are
detected.
# If set to INONLY_OR_INOUT, a LAN card will be considered down when its inbound
# message count stops increasing or when both inbound and outbound
# message counts stop increasing.
# If set to INOUT, both the inbound and outbound message counts must
# stop increasing before the card is considered down.
NETWORK_FAILURE_DETECTION    INOUT

# Package Configuration Parameters.
# Enter the maximum number of packages which will be configured in the cluster.
# You can not add packages beyond this limit.
# This parameter is required.
MAX_CONFIGURED_PACKAGES      150

# Access Control Policy Parameters.
#
# Three entries set the access control policy for the cluster:
# First line must be USER_NAME, second USER_HOST, and third USER_ROLE.
# Enter a value after each.
#
# 1. USER_NAME can either be ANY_USER, or a maximum of
#    8 login names from the /etc/passwd file on user host.
# 2. USER_HOST is where the user can issue Serviceguard commands.
#    If using Serviceguard Manager, it is the COM server.
#    Choose one of these three values: ANY_SERVICEGUARD_NODE, or
#    (any) CLUSTER_MEMBER_NODE, or a specific node. For node,
#    use the official hostname from domain name server, and not
#    an IP addresses or fully qualified name.
# 3. USER_ROLE must be one of these three values:
#    * MONITOR: read-only capabilities for the cluster and packages
#    * PACKAGE_ADMIN: MONITOR, plus administrative commands for packages
#      in the cluster
#    * FULL_ADMIN: MONITOR and PACKAGE_ADMIN plus the administrative
#      commands for the cluster.
#
# Access control policy does not set a role for configuration
# capability. To configure, a user must log on to one of the
# cluster's nodes as root (UID=0). Access control
# policy cannot limit root users' access.
#
# MONITOR and FULL_ADMIN can only be set in the cluster configuration file,
# and they apply to the entire cluster. PACKAGE_ADMIN can be set in the

```

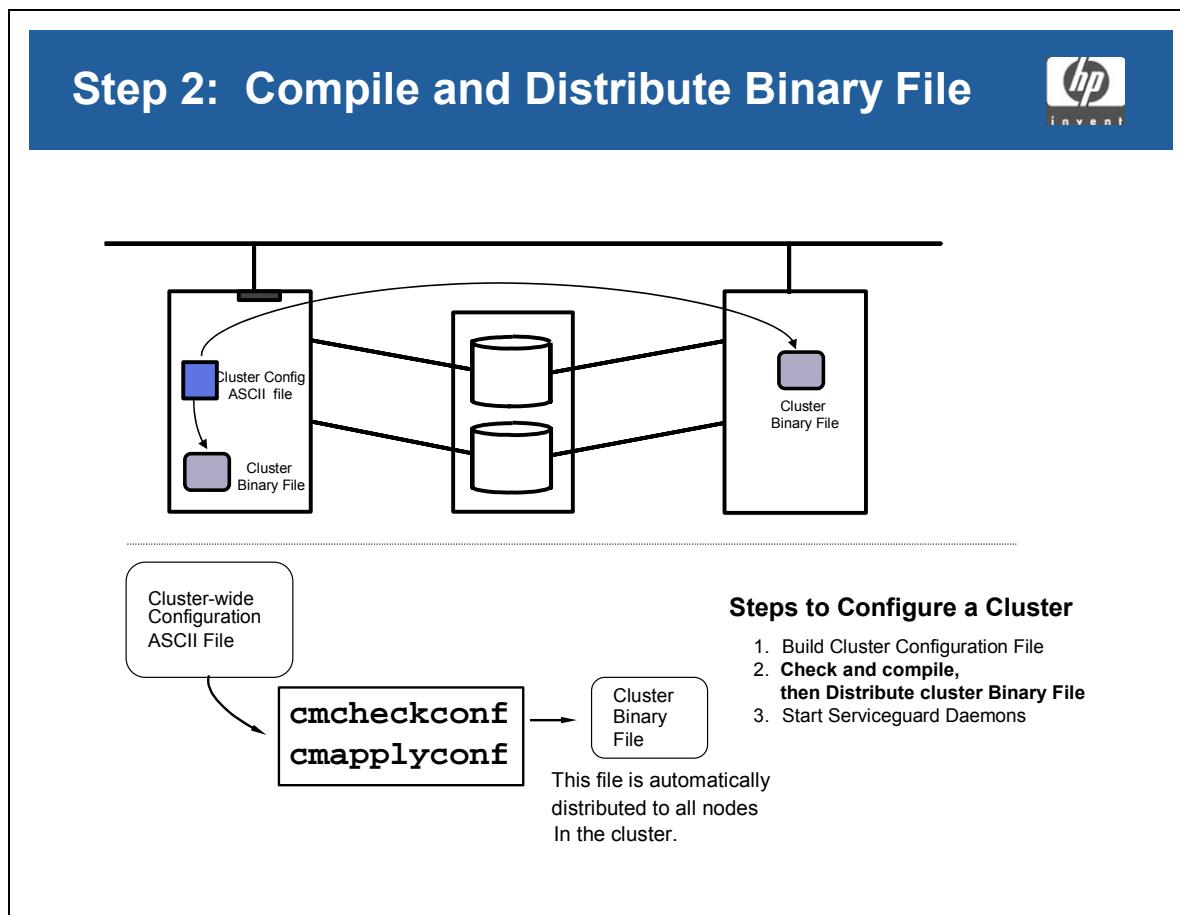
## Module 5

### Cluster Concepts and Configuration

```
# cluster or a package configuration file. If set in the cluster
# configuration file, PACKAGE_ADMIN applies to all configured packages.
# If set in a package configuration file, PACKAGE_ADMIN applies to that
# package only.
#
# Conflicting or redundant policies will cause an error while applying
# the configuration, and stop the process. The maximum number of access
# policies that can be configured in the cluster is 200.
#
# Example: to configure a role for user john from node noir to
# administer a cluster and all its packages, enter:
# USER_NAME    john
# USER_HOST    noir
# USER_ROLE    FULL_ADMIN

# List of cluster aware LVM Volume Groups. These volume groups will
# be used by package applications via the vgchange -a e command.
# Neither CVM or VxVM Disk Groups should be used here.
# For example:
# VOLUME_GROUP      /dev/vgdatabase
# VOLUME_GROUP      /dev/vg02
```

## 5-15. SLIDE: Step 2: Compile and Distribute Binary File



### Student Notes

Once the cluster configuration ASCII file is edited, the file should be checked for errors. The `cmcheckconf` command can be used to perform these checks:

```
# cmcheckconf -v -C cmclconfig.ascii
```

After the file has been verified as containing no errors, the `cmaplyconf` command is used to create and distribute the cluster binary file.

```
# cmaplyconf -v -C cmclconfig.ascii
```

See the following two pages for the actual screen output resulting from the `cmcheckconf` and `cmaplyconf` commands.

## 5-16. TEXT PAGE: The cmcheckconf Command

### Special Note

Here is the actual screen output resulting from the `cmcheckconf` command.  
This output was obtained while using version A.11.17 of Serviceguard.

```
root@rx26u230 [/etc/cmcluster]
# cmcheckconf -v -C cmclconfig.ascii
Checking cluster file: cmclconfig.ascii
Checking nodes ... Done
Checking existing configuration ... Done
Gathering storage information
Found 2 devices on node rx26u230
Found 2 devices on node rx26u231
Analysis of 4 devices should take approximately 1 seconds
0%----10%----20%----30%----40%----50%----60%----70%----80%----90%----100%
Found 2 volume groups on node rx26u230
Found 2 volume groups on node rx26u231
Analysis of 4 volume groups should take approximately 1 seconds
0%----10%----20%----30%----40%----50%----60%----70%----80%----90%----100%
Gathering network information
Beginning network probing (this may take a while)
Completed network probing
Checking for inconsistencies
Adding node rx26u230 to cluster cluster9
Adding node rx26u231 to cluster cluster9
cmcheckconf: Verification completed with no errors found.
Use the cmapplyconf command to apply the configuration.

root@rx26u230 [/etc/cmcluster]
#
```

## 5-17. TEXT PAGE: The cmapplyconf Command

### Special Note

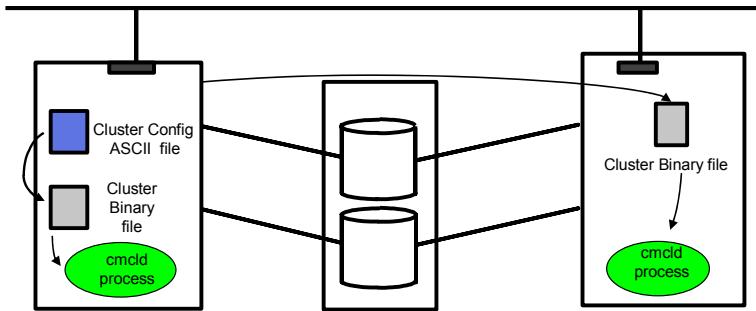
Here is the actual screen output resulting from the cmapplyconf command.  
This output was obtained while using version A.11.17 of Serviceguard.

```
root@rx26u230  [/etc/cmcluster]
# cmapplyconf -v -C cmclconfig.ascii
Checking cluster file: cmclconfig.ascii
Checking nodes ... Done
Checking existing configuration ... Done
Gathering storage information
Found 2 devices on node rx26u230
Found 2 devices on node rx26u231
Analysis of 4 devices should take approximately 1 seconds
0%----10%----20%----30%----40%----50%----60%----70%----80%----90%----100%
Found 2 volume groups on node rx26u230
Found 2 volume groups on node rx26u231
Analysis of 4 volume groups should take approximately 1 seconds
0%----10%----20%----30%----40%----50%----60%----70%----80%----90%----100%
Gathering network information
Beginning network probing (this may take a while)
Completed network probing
Checking for inconsistencies
Adding node rx26u230 to cluster cluster9
Adding node rx26u231 to cluster cluster9
Marking/unmarking volume groups for use in the cluster
Completed the cluster creation

root@rx26u230  [/etc/cmcluster]
#
```

## 5-18. SLIDE: Step 3: Start the Serviceguard Daemons

### Step 3: Start the Serviceguard Daemons



```
# cmrunc1 -v
```

#### Steps to Configure a Cluster

1. Build Cluster Configuration File
2. Compile and Distribute Binary File
- 3. Start the Serviceguard Daemons**

### Student Notes

Once the cluster binary file has been distributed to all nodes in the cluster, the `cmrunc1` command can be used to bring up the cluster.

```
# cmrunc1 -v
```

This command launches the appropriate Serviceguard daemons, including the `cmcld` daemon that monitors the cluster resources and sends or receives heartbeat packets.

---

## 5-19. TEXT PAGE: The cmrunc1 Command

### Special Note

Here is the actual screen output resulting from the `cmrunc1` command.  
This output was obtained while using version A.11.16 of Serviceguard.

```
root@rx26u230  [/etc/cmcluster]
# cmrunc1 -v
cmrunc1: Validating network configuration...
cmrunc1: Network validation complete
Waiting for cluster to form ..... done
Cluster successfully formed.
Check the syslog files on all nodes in the cluster to verify that no
warnings occurred during startup.

root@rx26u230  [/etc/cmcluster]
#
```

## 5-20. SLIDE: Cluster Configuration Procedure

### Cluster Configuration Procedure



1. Set up trusted hosts within cluster systems:

```
# vi /etc/cmcluster/cmclnodelist
```

this_node_hostname	root
other_node_hostname	root

2. # cd /etc/cmcluster
3. # cmquerycl -v [-q QS\_HOST] -C cmclconfig.ascii \  
-n node1 -n node2
4. # vi cmclconf.ascii  
    Modify CLUSTER\_NAME field  
    Modify timing parameters
5. # cmcheckconf -v -C /etc/cmcluster/cmclconf.ascii
6. # cmapplyconf -v -C /etc/cmcluster/cmclconf.ascii
7. # cmrunc1 -v
8. # cmviewcl -v

### Student Notes

The above slide summarizes the setup procedure for configuring a cluster. The details of each step are provided below:

#### (Step 1) Set Up Trusted Hosts

Prior to Serviceguard version A.11.16, the `/etc/cmcluster/cmclnodelist` file was used to configure all inter-host access for Serviceguard commands.

As of A.11.16, this file is only used to configure the cluster. Once the cluster is configured, an internal access list is used. Details will be discussed in a later module.

#### (Steps 2 – 4) Create the Cluster Configuration ASCII File

The cluster configuration ASCII file should be created in the `/etc/cmcluster` directory. The `cmquerycl` command can be used to gather the node-specific information and to create a template that can be easily customized for the cluster.

At the very least, the `CLUSTER_NAME` variable should be modified in the template. Other values should be modified as appropriate.

### (Steps 5 and 6) Compile and Distribute the Binary File

Once the cluster configuration ASCII file is edited, the file should be checked for syntax and logic errors. The `cmcheckconf` command performs the following checks:

- Network addresses and connections.
- Cluster lock disk connectivity (if using a cluster lock disk).
- Validity of configuration parameters for the cluster and packages.
- Uniqueness of names.
- Existence and permission of scripts specified in the command line.
- If all the nodes specified are in the same heartbeat subnet.
- If you specify the wrong configuration filename.
- If all nodes are accessible.
- No more than one CLUSTER\_NAME, HEARTBEAT\_INTERVAL, AUTO\_START\_TIMEOUT, and FIRST\_CLUSTER\_LOCK\_VG are specified.
- The value for HEARTBEAT\_INTERVAL  $\geq$  1 second; the value of NODE\_TIMEOUT is at least twice the value of HEARTBEAT\_INTERVAL.
- The value of AUTO\_START\_TIMEOUT are integers  $\geq$  0.
- At least 1 and no more than 16 NODE\_NAME's (Serviceguard OPS edition or Serviceguard) are specified.
- Each Node is connected to each heartbeat network.
- All heartbeat networks are of the same type of LAN.
- The network interface device files specified are valid LAN devices.
- There is no more than one serial ( RS232 ) port connection per node.
- VOLUME\_GROUP entries are not currently marked as cluster-aware.

After the file has been verified as containing no errors, the `cmaplyconf` command is used to create and distribute the cluster binary file.

### (Step 7) Bring Up the Cluster

The `cmruncl` command is used to bring up the cluster.

### (Step 8) Check the status of the cluster

```
# cmviewcl -v
```

## 5-21. SLIDE: Viewing the Cluster – cmviewcl Command

### Viewing the Cluster — cmviewcl Command



```
# cmviewcl -v

CLUSTER      STATUS
cluster9      up

NODE         STATUS      STATE
rx26u230    up          running

Cluster_Lock_LVM:
VOLUME_GROUP      PHYSICAL_VOLUME      STATUS
/dev/vg01        /dev/dsk/c20t0d7      up

Network_Parameters:
INTERFACE      STATUS      PATH      NAME
PRIMARY        up          0/1/2/0    lan1
STANDBY        up          0/2/1/0    lan2

NODE         STATUS      STATE
rx26u231    up          running

Cluster_Lock_LVM:
VOLUME_GROUP      PHYSICAL_VOLUME      STATUS
/dev/vg01        /dev/dsk/c20t0d7      up

Network_Parameters:
INTERFACE      STATUS      PATH      NAME
PRIMARY        up          0/1/2/0    lan1
STANDBY        up          0/2/1/0    lan2
```

### Student Notes

The `cmviewcl` command with the `-v` (verbose) option displays information about the status of the cluster and nodes within the cluster.

### Cluster States

The *status* of a cluster may be one of the following:

Up        At least one node has a running cluster daemon (`cmcld` ).

Down       No cluster daemons are running on any nodes.

Reforming   The cluster is in the process of determining its active membership.

### Node States

The *status* of a node is either up or down, depending on whether its cluster daemon is running or not. A node may also be in one of the following states:

Initializing       A node sees itself in this state after its daemon has started, but before it is ready to communicate with other nodes' daemons.

Failed	A node never sees itself in this state. Other active members of the cluster will see a node in this state.
Cluster reforming	A node in this state is running the protocols which ensure that all nodes agree to the new membership and the new coordinator of the cluster.
Running	A node in this state has completed all required activity for the last reformation and is operating normally.
Halted	A node never sees itself in this state. Other nodes will see it in this state after the node has gracefully left the cluster.

New in version A.11.17 is an option to `cmviewcl` to put the output in a format that is more easily parsed by a shell script.

```
# cmviewcl -f line

name=cluster9
id=1416255277
status=up
state=stable
coordinator=rx26u230
node:rx26u230|name=rx26u230
node:rx26u230|status=up
node:rx26u230|state=running
node:rx26u230|os_status=up
node:rx26u230|id=1
node:rx26u231|name=rx26u231
node:rx26u231|status=up
node:rx26u231|state=running
node:rx26u231|os_status=unknown
node:rx26u231|id=2
```

## 5-22. SLIDE: Checking the Cluster Log

### Checking the Cluster Log



```
# tail -f /var/adm/syslog/syslog.log
May 2 15:30:45 systemA cmcld[2048]: cmrunc1
May 2 15:30:47 systemA cmcld[2048]: Starting cluster protocols
May 2 15:30:48 systemA cmcld[2048]: Attempting to form a new cluster
May 2 15:30:50 systemA cmcld[2048]: Clearing the Cluster Lock
May 2 15:30:45 systemA cmcld[2048]: Turning on safety time protection
May 2 15:30:50 systemA cmcld[2048]: Two nodes have formed a new cluster
May 2 15:30:51 systemA cmcld[2048]: The new active membership is:
                                         systemA, systemB
May 2 15:30:52 systemA cmcld[2048]: clvmd initialized successfully
```

### Student Notes

The slide shows entries from the `/var/adm/syslog/syslog.log` file when the `cmrunc1` command was being executed.

In general, the `syslog.log` file is an excellent source of information regarding the activities that have taken place on the cluster as a whole. The `cmcld` daemon logs cluster messages to the HP-UX message daemon `syslogd`, using the daemon facility. Check `/etc/syslog.conf` for `syslogd` configuration on your systems.

## 5-23. LAB: Building a Cluster

### **Special Note:**

When building a cluster in this class, you must choose whether you wish to use a Quorum Server –or- an LVM Lock Disk. Please advise your instructor of your preference when asked.

#### **Quorum Server for Cluster Lock Services:**

If you wish to use a Quorum Server to provide cluster lock services, **notify your instructor immediately** and then turn the page and begin LAB 5-24.

#### **LVM Lock Disk:**

If you wish to use an LVM Lock Disk instead, turn to LAB 5-25 and begin configuring your cluster. Volume group `vg01`, previously configured in Module 4, is the volume group that will provide LVM Cluster Lock disk functionality.

---

## 5-24. LAB: Cluster Lock Using A Quorum Server

### Directions

Before you build your cluster, install the Quorum Server software on a system designated by the instructor. Then configure, startup, and test your quorum server. Then configure, startup, and test your cluster.

### Part 1: Identify the Depot location from which to swinstall the Quorum Server software

#### Special Note

In this lab, we will configure one quorum server for the entire class. Multiple Serviceguard clusters can all use the same Quorum Server.

1. Your instructor may ask for a volunteer to configure the Quorum Server. Ask your instructor for the name of the person who will complete all of Part 2 of this lab, "Configure the Quorum Server", and write that person's name here:

---

Name of person completing Part 2

2. Ask your instructor for the depot location of the Quorum Server software. If the software is located on a different server, note that as well. Write the depot location here:

---

Server and Depot Location

## Part 2: Configure the Quorum Server

1. Ask your instructor which system will be used as the Quorum Server. Write its hostname here:

## Quorum Server

The student whom the instructor designated to install the quorum server software should login to the quorum server now to perform the next several steps.

2. On the designated Quorum Server system, the student will install the Quorum Server software, product number B8467BA. Install this software using `depotserver` and the software depot (*noted above in Part I, item 2*). We install this software only on the system that will become our Quorum Server. The Quorum Server software is not installed on individual systems which later will be part of the cluster.

```
# swinstall -s depotserver:/pathname \*
```

- Now, with the software installed on the Quorum Server system, we need to create the authorization file to allow specific host systems to obtain quorum services. The *required* pathname for this authorization file is `/etc/cmcluster/qm_authfile`.

In this file, enter the hostnames of all cluster nodes that will access quorum services from this Quorum Server. Use one line per node, as in the following example:

r213c2  
r213c3  
r213c4  
r213c5

4. By default, quorum service run-time messages go to `stdout` and `stderr`. Instead, let's redirect these messages to a logfile. For this purpose, we will create the new subdirectory `/var/adm/qs`.
  
5. Now, edit the `/etc/inittab` file so that the Quorum Server will be started any time the system restarts or reboots. We also redirect `stdout` and `stderr` to a file in the `/var/adm/qs` subdirectory.

```
# vi /etc/inittab
```

and, at the bottom of the file, add a line that reads:

```
qs:345:respawn:/usr/lbin/qs >> /var/adm/qs/qs.log 2>&1
```

6. Next, start the Quorum Server software by telling `init` to re-read its configuration file.

```
# init q
```

7. Verify that the Quorum Server is running properly by checking the `/var/adm/qs/qs.log` file.

```
# cat /var/adm/qs/qs.log
```

The log file should contain entries that look something like the following, which indicates that the Quorum Server has properly started:

```
Nov 12 15:02:51:0: Starting Quorum Server
Nov 12 15:02:51:0: Total allocated: 466492 bytes, used: 20896 bytes, unused 429224 bytes
Nov 12 15:02:54:0: Server is up and waiting for connections at port 1238
```

## Part 3: Configure, Startup, and Test your Serviceguard Cluster

1. First, list the nodes that will comprise your Serviceguard cluster. Write their hostnames here:

\_\_\_\_\_  
Node1

\_\_\_\_\_  
Node2

2. Now, choose one node to be the "primary node" for your cluster. This is the node from which almost all cluster administration will be done. Write the hostname for your primary node here:

\_\_\_\_\_  
Primary Node

### Special Note

For the remainder of the week, your primary node will be the node that is generally used for all cluster administration, unless specifically mentioned otherwise.

3. Next, on both nodes, and before we start our actual configuration, edit the `/etc/lvmrc` file. We do this so that volume groups used within the cluster will be activated by Serviceguard. We do not want these volume groups activated by default whenever the node reboots. To accomplish this, edit the file and set the `AUTO_VG_ACTIVATE` line equal to 0 ( zero ).

AUTO\_VG\_ACTIVATE=0

4. Now, on your primary node, set up the special file used by Serviceguard that enables you to initially configure your cluster. We do this on the primary node by creating a new file called /etc/cmcluster/cmclnodelist. **Important:** All nodes in the cluster must be listed in this file. **The file is identical on all nodes**, and therefore we can easily `ftp` or `rcp` the file to every node in the cluster. The format for the file looks something like the following:

```
<Node1>      root  
<Node2>      root  
<Node3>      root  
<Node4>      root
```

5. Now, we create the cluster-wide ASCII configuration file. This file is where all cluster-wide information will be stored.

Note that we must specify: (a) the name of our Quorum Server, (b) the name of the cluster-wide ASCII file, and (c) the names of each node in our (*soon-to-be*) cluster.

6. Examine the cluster-wide ASCII configuration file built by the `cmquerycl` command *in the previous step*. Identify the following information:

```
# cd /etc/cmcluster  
# more cmclconfig.ascii
```

To what value is the name of the cluster (`"CLUSTER_NAME"`) set? \_\_\_\_\_

What is the hostname of the Quorum Server (`"QS_HOST"`)? \_\_\_\_\_

To what value is the Quorum Server polling interval (`"QS_POLLING_INTERVAL"`) set? \_\_\_\_\_

To what value is the HEARBEAT\_INTERVAL set? \_\_\_\_\_

To what value is the NODE\_TIMEOUT set? \_\_\_\_\_

To what value is the maximum number of packages (`"MAX_CONFIGURED_PACKAGES"` set)? \_\_\_\_\_

LAN Interfaces	Node1	Node2
Which lan interface card will be used as HEATBEAT_IP?		
Which lan interface card, if any, will be used as STATIONARY_IP?		
Which lan interface card, if any, will be used as a standby card (if you have an extra card)?		
Which lan interface card, if any, will be used as another standby card (if you have a second extra card)?		

Module 5  
**Cluster Concepts and Configuration**

7. Edit the cluster-wide ASCII configuration file to change some of these values (*seen above*):
  - a. Change your cluster name to whatever your instructor asks. Ask your instructor for this new name.
  - b. Set the QS\_POLLING\_INTERVAL to 120000000 microseconds (*120 seconds*).
  - c. Set the HEARTBEAT\_INTERVAL to 3000000 microseconds (*3 seconds*).
  - d. Set the NODE\_TIMEOUT value to 6000000 microseconds (*6 seconds*).
  - e. Set the AUTO\_START\_TIMEOUT value to 800000000 microseconds (*800 seconds*).
8. Let's add a non-root user to the list of users who can execute certain Serviceguard commands. In the cluster ascii file, give the `oracle` user Access Control rights that will enable him/her to monitor packages within the cluster. Allow this user to do this from either node in your cluster.
9. Now, on your primary node, use the `cmcheckconf` command to check the cluster-wide ASCII configuration file for errors. Fix any errors and re-check if necessary:
10. In order to see what files are created by the `cmapplyconf` command (*see step 11 below*), view the contents of the `/etc/cmcluster` directory as it is now, before we do step 10.
11. Next, use the `cmapplyconf` command to apply the configuration. The `cmapplyconf` command will always (1) compile the ASCII file to binary form, and (2) distribute this binary file to all nodes in the cluster.

12. Once again, view the contents of the `/etc/cmcluster` directory.

```
# ll /etc/cmcluster
```

Write the names of the file(s) created by the `cmapplyconf` command?

Files in the <code>/etc/cmcluster</code> directory		

13. Now, start cluster services on all nodes (*that is, start the cluster*). While it is a good practice to administer your cluster from one specific node, it is possible to start your cluster from any node (in the cluster).

14. Using the log file for `syslogd` (`/var/adm/syslog/syslog.log`), verify that the cluster has properly formed on all nodes in the cluster.

15. Now, view the complete status of the cluster. In the screen output, how many references do you see to "Quorum Server"?

Number of references seen to "Quorum Server" → \_\_\_\_\_

## Part 4: Test your Serviceguard Cluster

1. For our first test, from your primary node, halt the cluster.
  2. Now, with the cluster halted, shutdown and power off any one node except your primary node. In Part 4 of this lab, we will call this node (*the node to be shutdown*) "Node2".

Note that, if you are using remote equipment, the shutdown -hy 0 command will automatically cause a <power-off>.

3. Now, on your primary node, attempt to restart the cluster. Note that we are starting the cluster from the primary node only, but the `cmruncl` command will apply to all the nodes in the cluster.

As the cluster is trying to start, watch your screen carefully, and answer the following questions:

- a. Does the cluster form properly? Why or why not?
  
  - b. How can we restart the cluster if one node has become inoperable? See the manpage for `cmrunc1` and note the `-n` option. Note that it is possible to start the cluster with less than 100% node attendance.

```
# man cmrunc1
```

Start the cluster with this option.

4. Now, turn the power back on for Node2, and boot it up to run-level 3. If you are using remote equipment, issue a pc command from the Management Processor Command Menu.

Note: In order to interact with the system at the MP prompt, you will need to login to the MP on the system you shut down. If you do not know the MP hostname, ask your instructor.

To login to the MP, you'll need to supply an MP username and password. Once logged in, and at the MP prompt, type CM to enter the Command Menu. This is where you'll find the pc command, which is used to control power to your system.

After powering up the system, and while still at the MP :CM> prompt, type MA to return to the MP prompt. From here, type CO to access the system console and to login to unix once the system is back up.

**Answer:**

<power-on>

(Using local equipment)

or

MP:CM> pc

(Using remote equipment)

5. After Node2 is fully booted, check to see if it has automatically rejoined the cluster.

6. Examine the /etc/rc.config.d directory. Can you determine the name of the file in /etc/rc.config.d that controls whether or not a node automatically rejoins the cluster? **Hint:** you may be able to identify the file visually by listing the directory or, since all files in this directory are text files, a grep command might be handy.

```
# cd /etc/rc.config.d  
# ll
```

The name of this file is \_\_\_\_\_.

7. Since Node2 has not automatically rejoined the cluster (*step 5 above*), use the cmrundnode command to bring Node2 back into the running cluster. Afterward, check again to ensure that Node2 has rejoined the cluster and the cluster is now behaving correctly.

Congratulations on successfully starting and testing your Serviceguard cluster using a Quorum Server!

---

## 5-25. LAB: Cluster Lock Using LVM Disks

### Directions

In a previous module, we configured a shared volume group. Cluster Lock Disk functionality will be obtained from a disk in this volume group automatically. As such, if you elect to use an LVM Lock Disk in an I.T. environment, it is important to build the shared volume group first before building the cluster. Let's proceed with building our cluster.

### Part 1: Configuring your Serviceguard Cluster

1. First, list the nodes that will comprise your Serviceguard cluster. Write their hostnames here:

---

Node1

---

Node2

2. Now, choose one node to be the "primary node" for your cluster. This is the node from which almost all cluster administration will be done. Write the hostname for your primary node here:

---

Primary Node

#### Special Note

For the remainder of the week, your primary node will be the node that is generally used for all cluster administration, unless specifically mentioned otherwise.

3. Next, on both nodes, and before we start our actual configuration, edit the `/etc/lvmrc` file. We do this so that volume groups used within the cluster will be activated by Serviceguard. We do not want these volume groups activated by default whenever the node reboots. To accomplish this, edit the file and set the `AUTO_VG_ACTIVATE` line equal to 0 ( zero ).

---

`AUTO_VG_ACTIVATE=0`

Module 5  
**Cluster Concepts and Configuration**

4. Now, on your primary node, set up the special file used by Serviceguard that enables you to initially configure your cluster. We do this on the primary node by creating a new file named /etc/cmcluster/cmclnodelist. **Important:** All nodes in the cluster must be listed in this file. The file is identical on every node in our cluster, and therefore we can easily `ftp` or `rcp` the file to every node in the cluster. The format for the file looks something like the following:

Node1	root
Node2	root
Node3	root
Node4	root

5. Now, we create the cluster-wide ASCII configuration file. This file is where all cluster-wide information will be stored.

Note that we must specify the name of the cluster-wide ASCII file and the names of each node in our (*soon-to-be*) cluster.

6. Examine the cluster-wide ASCII configuration file built by the cmquerycl command (*in the previous step*). Identify the following information:

To what value is the name of the cluster ("CLUSTER\_NAME") set? \_\_\_\_\_

To what value is the HEARBEAT\_INTERVAL set? \_\_\_\_\_

To what value is the NODE\_TIMEOUT set? \_\_\_\_\_

To what value is maximum number of packages ("MAX\_CONFIGURED\_PACKAGES") set? \_\_\_\_\_

LAN Interfaces	Node1	Node2
Which lan interface card will be used as HEATBEAT_IP?		
Which lan interface card, if any, will be used as STATIONARY_IP?		
Which lan interface card, if any, will be used as a standby card ( if you have an extra card )?		
Which lan interface card, if any, will be used as another standby card ( if you have a second extra card )?		

Module 5  
**Cluster Concepts and Configuration**

7. Edit the cluster-wide ASCII configuration file to change some of these values (*seen above*):
  - a. Change your cluster name to whatever your instructor asks. Ask your instructor for this new name and write it here \_\_\_\_\_.
  - b. Set the HEARTBEAT\_INTERVAL to 3000000 microseconds (*3 seconds*).
  - c. Set the NODE\_TIMEOUT value to 6000000 microseconds (*6 seconds*).
  - d. Set the AUTO\_START\_TIMEOUT value to 800000000 microseconds (*800 seconds*).
8. Let's add a non-root user to the list of users who can execute certain Serviceguard commands. In the cluster ascii file, give the `oracle` user Access Control rights that will enable him/her to monitor packages within the cluster. Allow this user to do this from either node in your cluster.
9. Now, on your primary node, use the `cmcheckconf` command to check the cluster-wide ASCII configuration file for errors. Fix any errors and re-check if necessary:
10. To see what files are created by the `cmapplyconf` command (*see step 11 below*), view the contents of the `/etc/cmcluster` directory as it is now, before we do step 10.

11. Next, still on your primary node, use the `cmapplyconf` command. The `cmapplyconf` command will always: (a) compile the `cmclconfig.ascii` file to binary, and (b) distribute the binary file to all nodes.

12. Once again, view the contents of the `/etc/cmcluster` directory.  
Write the names of the file(s) created by the `cmapplyconf` command?

Files in the <code>/etc/cmcluster</code> directory		

13. Now, start cluster services on all nodes (*that is, start the cluster*). While it is a good practice to administer your cluster from one specific node, it is possible to start your cluster from any node (in the cluster).
14. Using the log file for `syslogd` (`/var/adm/syslog/syslog.log`), verify that the cluster has properly formed on all nodes in the cluster.
15. Using `cmviewcl -v`, verify that the cluster has properly formed on all nodes in the cluster.

## **Part 2: Test your Serviceguard Cluster**

1. For our first test, from your primary node, halt the cluster.
2. Next, with the cluster halted, shutdown and power off any one node except your primary node. In this part of the lab, we will call this node "Node2" (*the node to be shutdown*).

Note that, if you are using remote equipment, the shutdown -hy 0 command will automatically cause a <power-off>.

3. Now, on your primary node, attempt to restart the cluster. Note that we are starting the cluster from the primary node only, but the `cmrunc1` command will apply to all the nodes in the cluster.

As the cluster is trying to start, watch your screen carefully, and answer the following questions:

- a. Does the cluster form properly? Why or why not?
- b. How can we restart the cluster if one node has become inoperable? See the manpage for `cmrunc1` and note the `-n` option. Note that it is possible to start the cluster with less than 100% node attendance.

```
# man cmrunc1
```

Start the cluster with this option.

4. Now turn the power back on for Node2, then boot Node2 up to run-level 3. If you are using remote equipment, issue a pc command from the Management Processor Command menu.

Note: In order to interact with the system at the MP prompt, you will need to login to the MP on the system you shut down. If you do not know the MP hostname, ask your instructor.

To login to the MP, you'll need to supply an MP username and password. Once logged in, and at the MP prompt, type CM to enter the Command Menu. This is where you'll find the pc command, which is used to control power to your system.

After powering up the system, and while still at the MP:CM> prompt, type MA to return to the MP prompt. From here, type CO to access the system console and to login to unix once the system is back up.

- After Node2 is fully booted, check to see if Node2 has automatically rejoined the cluster?
  - Now, examine the /etc/rc.config.d directory. Can you discover the name of the file in /etc/rc.config.d that controls whether or not a node automatically rejoins the cluster? **Hint:** you may be able to identify the file visually by listing the directory or, since all files in this directory are text files, a grep command might be handy.

The name of this file is .

## Cluster Concepts and Configuration

7. Since `Node2` has not automatically rejoined the cluster (*item 5 above*), use the `cmrundnode` command to bring `Node2` back into the running cluster. Afterward, check again to ensure that `Node2` has rejoined the cluster and the cluster is now behaving correctly.

Congratulations on successfully configuring, starting, and testing your Serviceguard cluster using an LVM Lock Disk.

---

## 5-26. LAB Solution: Cluster Lock Using A Quorum Server

### Directions

Before you build your cluster, install the Quorum Server software on a system designated by the instructor. Then configure, startup, and test your quorum server. Then configure, startup, and test your cluster.

#### Part 1: Identify the Depot location from which to swinstall the Quorum Server software

**Special Note**

In this lab, we will configure one quorum server for the entire class. Multiple Serviceguard clusters can all use the same Quorum Server.

1. Your instructor may ask for a volunteer to configure the Quorum Server. Ask your instructor for the name of the person who will complete all of Part 2 of this lab, "Configure the Quorum Server", and write that person's name here:

---

Name of person completing Part 2

2. Ask your instructor for the depot location of the Quorum Server software. If the software is located on a different server, note that as well. Write the depot location here:

---

Server and Depot Location

## Part 2: Configure the Quorum Server

1. Ask your instructor which system will be used as the Quorum Server. Write its hostname here:

---

Quorum Server

The student whom the instructor designated to install the quorum server software should login to the quorum server now to perform the next several steps.

2. On the designated Quorum Server system, the student will install the Quorum Server software, product number B8467BA. Install this software using `depotserver` and the software depot (*noted above in Part I, item 2*). We install this software only on the system that will become our Quorum Server. The Quorum Server software is not installed on individual systems which later will be part of the cluster.

```
# swinstall -s depotserver:/pathname \*
```

3. Now, with the software installed on the Quorum Server system, we need to create the authorization file to allow specific host systems to obtain quorum services. The *required* pathname for this authorization file is `/etc/cmcluster/qs_authfile`.

In this file, enter the hostnames of all cluster nodes that will access quorum services from this Quorum Server. Use one line per node, as in the following example:

```
r213c2
r213c3
r213c4
r213c5
```

**Answer:**

```
# cd /etc/cmcluster
# vi qs_authfile
```

4. By default, quorum service run-time messages go to `stdout` and `stderr`. Instead, let's redirect these messages to a logfile. For this purpose, we will create the new subdirectory `/var/adm/qs`.

**Answer:**

```
# mkdir /var/adm/qs
```

5. Now, edit the `/etc/inittab` file so that the Quorum Server will be started any time the system restarts or reboots. We also redirect `stdout` and `stderr` to a file in the `/var/adm/qs` subdirectory.

```
# vi /etc/inittab
```

and, at the bottom of the file, add a line that reads:

```
qs:345:respawn:/usr/lbin/qs  >>  /var/adm/qs/qs.log  2>&1
```

6. Next, start the Quorum Server software by telling `init` to re-read its configuration file.

```
# init q
```

7. Verify that the Quorum Server is running properly by checking the `/var/adm/qs/qs.log` file.

```
# cat /var/adm/qs/qs.log
```

The log file should contain entries that look something like the following, which indicates that the Quorum Server has properly started:

```
Nov 12 15:02:51:0: Starting Quorum Server
Nov 12 15:02:51:0: Total allocated: 466492 bytes, used: 20896 bytes, unused 429224 bytes
Nov 12 15:02:54:0: Server is up and waiting for connections at port 1238
```

## Part 3: Configure, Startup, and Test your Serviceguard Cluster

1. First, list the nodes that will comprise your Serviceguard cluster. Write their hostnames here:

\_\_\_\_\_  
Node1

\_\_\_\_\_  
Node2

2. Now, choose one node to be the "primary node" for your cluster. This is the node from which almost all cluster administration will be done. Write the hostname for your primary node here:

\_\_\_\_\_  
Primary Node

### Special Note

For the remainder of the week, your primary node will be the node that is generally used for all cluster administration, unless specifically mentioned otherwise.

3. Next, on both nodes, and before we start our actual configuration, edit the /etc/lvmrc file. We do this so that volume groups used within the cluster will be activated by Serviceguard. We do not want these volume groups activated by default whenever the node reboots. To accomplish this, edit the file and set the AUTO\_VG\_ACTIVATE line equal to 0 ( zero ).

AUTO\_VG\_ACTIVATE=0

### Answer:

```
# vi /etc/lvmrc
```

- Now, on your primary node, set up the special file used by Serviceguard that enables you to initially configure your cluster. We do this on the primary node by creating a new file called `/etc/cmcluster/cmclnodelist`. **Important:** All nodes in the cluster must be listed in this file. **The file is identical on all nodes**, and therefore we can easily `ftp` or `rcp` the file to every node in the cluster. The format for the file looks something like the following:

<Node1>	root
<Node2>	root
<Node3>	root
<Node4>	root

**Answer:**

```
# cd /etc/cmcluster  
# vi cmclnodelist  
# chmod 444 cmclnodelist  
# rcp cmclnodelist <Node2>:/etc/cmcluster/. (to transfer the file to Node2  
and also to all other nodes)
```

- Now, we create the cluster-wide ASCII configuration file. This file where all cluster-wide information will be stored.

Note that we must specify: (a) the name of our Quorum Server, (b) the name of the cluster-wide ASCII file, and (c) the names of each node in our (*soon-to-be*) cluster.

**Answer:**

```
# cd /etc/cmcluster  
# cmquerycl -v -q quorum_server -C cmclconfig.ascii -n <Node1> \  
-n <Node2>
```

Module 5  
**Cluster Concepts and Configuration**

6. Examine the cluster-wide ASCII configuration file built by the `cmquerycl` command *in the previous step*. Identify the following information:

```
# cd /etc/cmcluster  
# more cmclconfig.ascii
```

To what value is the name of the cluster (`"CLUSTER_NAME"`) set? \_\_\_\_\_

What is the hostname of the Quorum Server (`"QS_HOST"`)? \_\_\_\_\_

To what value is the Quorum Server polling interval (`"QS_POLLING_INTERVAL"`) set? \_\_\_\_\_

To what value is the HEARBEAT\_INTERVAL set? \_\_\_\_\_

To what value is the NODE\_TIMEOUT set? \_\_\_\_\_

To what value is the maximum number of packages (`"MAX_CONFIGURED_PACKAGES"`) set? \_\_\_\_\_

LAN Interfaces	Node1	Node2
Which lan interface card will be used as HEATBEAT_IP?		
Which lan interface card, if any, will be used as STATIONARY_IP?		
Which lan interface card, if any, will be used as a standby card (if you have an extra card)?		
Which lan interface card, if any, will be used as another standby card (if you have a second extra card)?		

7. Edit the cluster-wide ASCII configuration file to change some of these values (*seen above*):
    - a. Change your cluster name to whatever your instructor asks. Ask your instructor for this new name.
    - b. Set the QS\_POLLING\_INTERVAL to 120000000 microseconds (*120 seconds*).
    - c. Set the HEARTBEAT\_INTERVAL to 3000000 microseconds (*3 seconds*).
    - d. Set the NODE\_TIMEOUT value to 6000000 microseconds (*6 seconds*).
    - e. Set the AUTO\_START\_TIMEOUT value to 800000000 microseconds (*800 seconds*).

**Answer:**

```
# vi /etc/cmcluster/cmclconfig.ascii
```

- Let's add a non-root user to the list of users who can execute certain Serviceguard commands. In the cluster ascii file, give the `oracle` user Access Control rights that will enable him/her to monitor packages within the cluster. Allow this user to do this from either node in your cluster.

## Answer:

```
# vi /etc/cmcluster/cmclconfig.ascii and add the following:
```

```
USER_NAME    oracle
USER_HOST    CLUSTER_MEMBER_NODE
USER_ROLE    MONITOR
```

- Now, on your primary node, use the `cmcheckconf` command to check the cluster-wide ASCII configuration file for errors. Fix any errors and re-check if necessary:

### **Answer:**

```
# cmcheckconf -v -C cmclconfig.ascii
```

10. In order to see what files are created by the `cmapplyconf` command (see step 11 below), view the contents of the `/etc/cmcluster` directory as it is now, before we do step 10.

### **Answer:**

```
# cd /etc/cmcluster  
# ll
```

Module 5

**Cluster Concepts and Configuration**

11. Next, use the `cmaplyconf` command to apply the configuration. The `cmaplyconf` command will always (1) compile the ASCII file to binary form, and (2) distribute this binary file to all nodes in the cluster.

**Answer:**

```
# cd /etc/cmcluster  
# cmaplyconf -v -C cmclconfig.ascii
```

12. Once again, view the contents of the `/etc/cmcluster` directory.

```
# ll /etc/cmcluster
```

Write the names of the file(s) created by the `cmaplyconf` command?

Files in the <code>/etc/cmcluster</code> directory		

13. Now, start cluster services on all nodes (*that is, start the cluster*). While it is a good practice to administer your cluster from one specific node, it is possible to start your cluster from any node (in the cluster).

**Answer:**

```
# cmrunc1 -v
```

14. Using the log file for `syslogd` (`/var/adm/syslog/syslog.log`), verify that the cluster has properly formed on all nodes in the cluster.

**Answer:**

```
# cd /var/adm/syslog  
# tail -75 syslog.log | more
```

15. Now, view the complete status of the cluster. In the screen output, how many references do you see to "Quorum Server"?

**Answer:**

```
# cmviewcl -v
```

Number of references seen to "Quorum Server" → \_\_\_\_\_

## Part 4: Test your Serviceguard Cluster

1. For our first test, from your primary node, halt the cluster.

**Answer:**

```
# cmhaltcl -v
```

2. Now, with the cluster halted, shutdown and power off any one node except your primary node. In Part 4 of this lab, we will call this node (*the node to be shutdown*) "Node2".

Note that, if you are using remote equipment, the `shutdown -hy 0` command will automatically cause a <power-off>.

**Answer:**

```
# cd /                                     (on Node2)
# shutdown -hy 0                           (on Node2)
< power off >                         (on Node2)
```

3. Now, on your primary node, attempt to restart the cluster. Note that we are starting the cluster from the primary node only, but the `cmrunc1` command will apply to all the nodes in the cluster.

As the cluster is trying to start, watch your screen carefully, and answer the following questions:

- c. Does the cluster form properly? Why or why not?

**Answer:**

No. We do not have 100% of the cluster nodes.

- d. How can we restart the cluster if one node has become inoperable? See the manpage for `cmrunc1` and note the `-n` option. Note that it is possible to start the cluster with less than 100% node attendance.

```
# man cmrunc1
```

Start the cluster with this option.

**Answer:**

```
# cmrunc1 -v -n <Node1>
```

4. Now, turn the power back on for Node2 , and boot it up to run-level 3. If you are using remote equipment, issue a pc command from the Management Processor Command Menu.

Note: In order to interact with the system at the MP prompt, you will need to login to the MP on the system you shut down. If you do not know the MP hostname, ask your instructor.

To login to the MP, you'll need to supply an MP username and password. Once logged in, and at the MP prompt, type CM to enter the Command Menu. This is where you'll find the pc command, which is used to control power to your system.

After powering up the system, and while still at the MP :CM> prompt, type MA to return to the MP prompt. From here, type CO to access the system console and to login to unix once the system is back up.

**Answer:**

<power-on>

(Using local equipment)

or

MP:CM> pc

(Using remote equipment)

5. After Node2 is fully booted, check to see if it has automatically rejoined the cluster.

**Answer:**

```
# cmviewcl -v
```

6. Examine the /etc/rc.config.d directory. Can you determine the name of the file in /etc/rc.config.d that controls whether or not a node automatically rejoins the cluster? **Hint:** you may be able to identify the file visually by listing the directory or, since all files in this directory are text files, a grep command might be handy.

```
# cd /etc/rc.config.d
# ll
```

The name of this file is \_\_\_\_\_.

**Answer:**

```
# cd /etc/rc.config.d
# ll
# grep -i cluster *
```

Module 5  
**Cluster Concepts and Configuration**

7. Since Node2 has not automatically rejoined the cluster (*step 5 above*), use the cmrunnode command to bring Node2 back into the running cluster. Afterward, check again to ensure that Node2 has rejoined the cluster and the cluster is now behaving correctly.

**Answer:**

```
# cmrunnode -v <Node2>
# cmviewcl -v
```

Congratulations on successfully starting and testing your Serviceguard cluster using a Quorum Server!

---

## 5-27. LAB Solution: Cluster Lock Using LVM Disks

### Directions

In a previous module, we configured a shared volume group. Cluster Lock Disk functionality will be obtained from a disk in this volume group automatically. As such, if you elect to use an LVM Lock Disk in an I.T. environment, it is important to build the shared volume group first before building the cluster. Let's proceed with building our cluster.

### Part 1: Configuring your Serviceguard Cluster

1. First, list the nodes that will comprise your Serviceguard cluster. Write their hostnames here:

---

Node1

---

Node2

2. Now, choose one node to be the "primary node" for your cluster. This is the node from which almost all cluster administration will be done. Write the hostname for your primary node here:

---

Primary Node

#### Special Note

For the remainder of the week, your primary node will be the node that is generally used for all cluster administration, unless specifically mentioned otherwise.

3. Next, on both nodes, and before we start our actual configuration, edit the /etc/lvmrc file. We do this so that volume groups used within the cluster will be activated by Serviceguard. We do not want these volume groups activated by default whenever the node reboots. To accomplish this, edit the file and set the AUTO\_VG\_ACTIVATE line equal to 0 ( zero ).

---

AUTO\_VG\_ACTIVATE=0

#### Answer:

```
# vi /etc/lvmrc
```

Module 5  
**Cluster Concepts and Configuration**

4. Now, on your primary node, set up the special file used by Serviceguard that enables you to initially configure your cluster. We do this on the primary node by creating a new file named /etc/cmcluster/cmclnodelist. **Important:** All nodes in the cluster must be listed in this file. The file is identical on every node in our cluster, and therefore we can easily `ftp` or `rcp` the file to every node in the cluster. The format for the file looks something like the following:

Node1	root
Node2	root
Node3	root
Node4	root

**Answer:**

```
# cd /etc/cmcluster
# vi cmclnodelist
# chmod 444 cmclnodelist
# rcp cmclnodelist Node2:/etc/cmcluster/.      (to transfer the file to Node2
                                                and also to all other nodes)
```

5. Now, we create the cluster-wide ASCII configuration file. This file is where all cluster-wide information will be stored.

Note that we must specify the name of the cluster-wide ASCII file and the names of each node in our (*soon-to-be*) cluster.

**Answer:**

```
# cd /etc/cmcluster
# cmquerycl -v -C cmclconfig.ascii -n <Node1> -n <Node2>
```

6. Examine the cluster-wide ASCII configuration file built by the `cmquerycl` command (*in the previous step*). Identify the following information:

To what value is the name of the cluster ("CLUSTER\_NAME") set? \_\_\_\_\_

To what value is the HEARBEAT\_INTERVAL set? \_\_\_\_\_

To what value is the NODE\_TIMEOUT set? \_\_\_\_\_

To what value is maximum number of packages ("MAX\_CONFIGURED\_PACKAGES") set? \_\_\_\_\_

LAN Interfaces	Node1	Node2
Which lan interface card will be used as HEATBEAT_IP?		
Which lan interface card, if any, will be used as STATIONARY_IP?		
Which lan interface card, if any, will be used as a standby card ( if you have an extra card )?		
Which lan interface card, if any, will be used as another standby card ( if you have a second extra card )?		

**Answer:**

```
# cd /etc/cmcluster
# more cmclconfig.ascii
```

Module 5  
**Cluster Concepts and Configuration**

7. Edit the cluster-wide ASCII configuration file to change some of these values (*seen above*):

- a. Change your cluster name to whatever your instructor asks. Ask your instructor for this new name and write it here \_\_\_\_\_.
- b. Set the HEARTBEAT\_INTERVAL to 3000000 microseconds (*3 seconds*).
- c. Set the NODE\_TIMEOUT value to 6000000 microseconds (*6 seconds*).
- d. Set the AUTO\_START\_TIMEOUT value to 800000000 microseconds (*800 seconds*).

**Answer:**

```
# cd /etc/cmcluster
# vi cmclconfig.ascii
```

8. Let's add a non-root user to the list of users who can execute certain Serviceguard commands. In the cluster ascii file, give the oracle user Access Control rights that will enable him/her to monitor packages within the cluster. Allow this user to do this from either node in your cluster.

**Answer:**

```
# vi /etc/cmcluster/cmclconfig.ascii and add the following:
```

```
USER_NAME    oracle
USER_HOST    CLUSTER_MEMBER_NODE
USER_ROLE    MONITOR
```

9. Now, on your primary node, use the `cmcheckconf` command to check the cluster-wide ASCII configuration file for errors. Fix any errors and re-check if necessary:

**Answer:**

```
# cd /etc/cmcluster
# cmcheckconf -v -C cmclconfig.ascii
```

10. To see what files are created by the `cmapplyconf` command (*see step 10 below*), view the contents of the `/etc/cmcluster` directory as it is now, before we do step 10.

**Answer:**

```
# cd /etc/cmcluster
# ll
```

11. Next, still on your primary node, use the `cmaplyconf` command. The `cmaplyconf` command will always: (a) compile the `cmclconfig.ascii` file to binary, and (b) distribute the binary file to all nodes.

**Answer:**

```
# cd /etc/cmcluster
# cmaplyconf -v -C cmclconfig.ascii
```

12. Once again, view the contents of the `/etc/cmcluster` directory.  
Write the names of the file(s) created by the `cmaplyconf` command?

**Answer:**

```
# cd /etc/cmcluster
# ll
```

Files in the <code>/etc/cmcluster</code> directory		

13. Now, start cluster services on all nodes (*that is, start the cluster*). While it is a good practice to administer your cluster from one specific node, it is possible to start your cluster from any node (in the cluster).

**Answer:**

```
# cmrunc1 -v
```

Module 5  
**Cluster Concepts and Configuration**

14. Using the log file for `syslogd` ( `/var/adm/syslog/syslog.log` ), verify that the cluster has properly formed on all nodes in the cluster.

**Answer:**

```
# tail -75 /var/adm/syslog/syslog.log | more
```

15. Using `cmviewcl -v`, verify that the cluster has properly formed on all nodes in the cluster.

**Answer:**

```
# cmviewcl -v
```

## Part 2: Test your Serviceguard Cluster

1. For our first test, from your primary node, halt the cluster.

**Answer:**

```
# cmhaltcl -v
```

2. Next, with the cluster halted, shutdown and power off any one node except your primary node. In this part of the lab, we will call this node "Node2" (*the node to be shutdown*).

Note that, if you are using remote equipment, the shutdown -hy 0 command will automatically cause a <power-off>.

**Answer:**

```
# cd /                                     (on Node2)  
# shutdown -hy 0                           (on Node2)  
<power-off>                            (on Node2)
```

3. Now, on your primary node, attempt to restart the cluster. Note that we are starting the cluster from the primary node only, but the `cmrunc1` command will apply to all the nodes in the cluster.

As the cluster is trying to start, watch your screen carefully, and answer the following questions:

- a. Does the cluster form properly? Why or why not?

**Answer:**

No. We do not have 100% of the cluster nodes.

- b. How can we restart the cluster if one node has become inoperable? See the manpage for `cmrunc1` and note the `-n` option. Note that it is possible to start the cluster with less than 100% node attendance.

```
# man cmrunc1
```

Start the cluster with this option.

```
# cmrunc1 -v -n <Node1>
```

Module 5  
**Cluster Concepts and Configuration**

4. Now turn the power back on for Node2, then boot Node2 up to run-level 3. If you are using remote equipment, issue a pc command from the Management Processor Command menu.

Note: In order to interact with the system at the MP prompt, you will need to login to the MP on the system you shut down. If you do not know the MP hostname, ask your instructor.

To login to the MP, you'll need to supply an MP username and password. Once logged in, and at the MP prompt, type CM to enter the Command Menu. This is where you'll find the pc command, which is used to control power to your system.

After powering up the system, and while still at the MP:CM> prompt, type MA to return to the MP prompt. From here, type CO to access the system console and to login to unix once the system is back up.

**Answer:**

<power-on> (Using local equipment)

or

MP:CM> pc (Using remote equipment)

5. After Node2 is fully booted, check to see if Node2 has automatically rejoined the cluster?

**Answer:**

# cmviewcl -v

6. Now, examine the /etc/rc.config.d directory. Can you discover the name of the file in /etc/rc.config.d that controls whether or not a node automatically rejoins the cluster? **Hint:** you may be able to identify the file visually by listing the directory or, since all files in this directory are text files, a grep command might be handy.

The name of this file is \_\_\_\_\_.

**Answer:**

```
# cd /etc/rc.config.d
# ll
# grep -i cluster *      (the file is /etc/rc.config.d/cmcluster)
```

7. Since Node2 has not automatically rejoined the cluster (*item 5 above*), use the `cmanrunnode` command to bring Node2 back into the running cluster. Afterward, check again to ensure that Node2 has rejoined the cluster and the cluster is now behaving correctly.

## **Answer:**

```
# cmrunnode -v <node2>  
# cmviewcl -v
```

Congratulations on successfully configuring, starting, and testing your Serviceguard cluster using an LVM Lock Disk.

Module 5  
**Cluster Concepts and Configuration**

---

# Module 6 — Additional Cluster Features

## Objectives

Upon completion of this module, you will be able to do the following:

- Mark a volume group as belonging to a Serviceguard cluster.
- Activate a shared volume group in an exclusive read/write access mode.
- Configure and test the local LAN failover capability of the Serviceguard cluster.
- Describe the differences between heartbeat LAN cards and standby LAN cards.
- List two methods in which a node can *join* an existing cluster.
- List two methods in which a node can *leave* an existing cluster.

## **6-1. SLIDE: Additional Cluster Features**

### **Additional Cluster Features**



**In this module, we will discuss in detail ...**

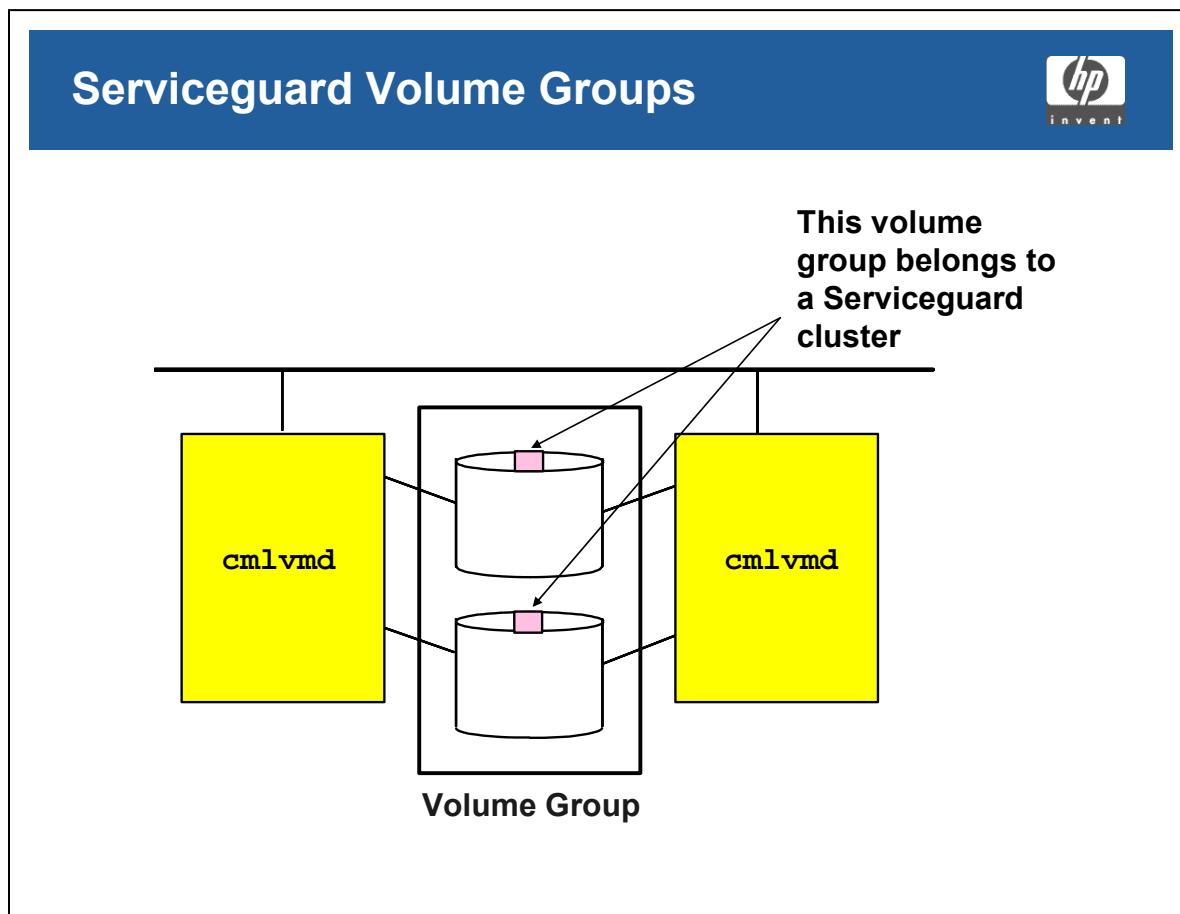
- Serviceguard Volume Groups
- Node Failures, Node Exits, and Node Joins
- Local LAN Card Failover

### **Student Notes**

This module focuses on additional features of a Serviceguard cluster which have not yet been highlighted. These features include:

- LVM enhancements specifically for Serviceguard that greatly improve the integrity data stored in a Serviceguard high availability environment.
- Commands that allow control over node "exits" and node "joins", which provide flexibility when managing a Serviceguard cluster.
- Local LAN Card failover which provides redundancy protection for the local LAN cards.

## 6-2. SLIDE: Serviceguard Volume Groups



### Student Notes

When LVM was designed in the late 1980s, the situation in which two systems would be cabled to the same volume group was not considered. As a result, if two systems cabled to the same volume group try to access the shared volume group concurrently using standard LVM functionality, LVM will allow it.

This means that both systems could activate the volume group concurrently using standard LVM. Both systems could mount the shared file system concurrently using standard LVM, and both systems could try to create a file in the exact same spot of the file system concurrently using standard LVM. Because each system contains its own view of the file system and that view would be written out to disk every 30 seconds by flushing the buffer cache, the shared file system would become corrupted very quickly.

To remedy this situation, the standard LVM functionality was modified to indicate when a volume group belonged to a Serviceguard cluster. If a volume group is part of a Serviceguard cluster, only one node will be allowed to access the volume group at a time. If a second system tries to access the volume group while the first system has it activated, an error will be returned to the second system indicating that the volume group is already active by another node in the cluster.

### **How Do Serviceguard Volume Groups Work?**

In the LVM header structure, a single bit (which was previously unused with standard LVM) is turned on to indicate membership with a Serviceguard cluster. If the bit is turned on, the volume group is assumed to be part of a Serviceguard cluster and only one node will be allowed to activate the volume group at a time.

In addition, the cluster ID is written to the volume group headers so that no other cluster can activate the volume group.

The enforcement of single node access to a volume group is done through the `cmlvmd` daemon process. This process must always be running on every node while the cluster is active. When a node tries to activate a volume group with the `vgchange` command, the request is passed to the `cmlvmd` daemon for approval. The daemon determines activation by broadcasting a message to all other `cmlvmd` daemon processes on other systems, asking if they currently have the volume group in question active. If a node already has the volume group active, then its corresponding `cmlvmd` daemon will respond indicating the volume group is already active, causing the attempted activation request to fail.

A volume group can only be activated if all other `cmlvmd` daemons respond with a status of not active for the volume group attempting to be activated.

## 6–3. SLIDE: Marking Volume Groups for use in Serviceguard

### Marking Volume Groups for use in Serviceguard



Marking Volume Group  
for Serviceguard

`vgchange -c y VGName`

Marking Volume Group  
as non-Serviceguard

`vgchange -c n VGName`

Standard Volume Group  
Activation ( classical LVM )

`vgchange -a y VGName`

Standard Volume Group  
Deactivation ( classical LVM )

`vgchange -a n VGName`

Exclusive Volume Group  
Activation

`vgchange -a e VGName`

Exclusive Volume Group  
Deactivation ( same as classical )

`vgchange -a n VGName`

### Student Notes

New arguments and options have been added to the `vgchange` command to support Serviceguard marked volume groups.

### Marking a Volume Group for Serviceguard

The `-c` option has been added to the `vgchange` command to enable you to mark a volume group for clustering. When a volume group is marked for clustering, only one node in the cluster at a time will be allowed to activate the cluster.

The `vgchange -c y VGName` command marks a volume group as part of a cluster. The `vgchange -c n VGName` command removes an existing mark, indicating that the volume group is no longer part of a cluster. These options are applied automatically by the `cmaplyconf` command when the volume group is listed in the cluster-wide ASCII file.

### Volume Group Activation Modes

There are two ways to activate a volume group for write access: shared access and exclusive access:

**Shared Read/Write Activation**

The vgchange -a y VGName command activates a volume group for *shared* read/write access.

When activated in shared read/write mode, both nodes can write to the file system concurrently. This is standard LVM functionality. This mode of activation is not supported for Serviceguard volume groups. This mode of activation can only be used on non-marked volume groups.

**Exclusive Read/Write Activation**

The vgchange -a e VGName command activates a volume group for *exclusive* read/write access.

When a node activates a volume group in exclusive read/write mode, the node will be the only node to have write access to the volume group. This mode of activation only works if no other node in the cluster has already activated the volume group. This mode of activation is the only supported activation mode for Serviceguard marked volume groups.

## 6-4. SLIDE: Exclusive Mode Volume Group Activation

### Exclusive Mode Volume Group Activation

The diagram shows two nodes, each with a yellow box labeled 'cmlvmd'. They are connected to a central 'Volume Group' which contains two cylinders representing disks. A question bubble above the left node asks, 'Q: Do you have the Volume Group activated?' and an answer bubble above the right node responds, 'A: No, I do not. Go ahead and activate it on your side.'

### Student Notes

Exclusive volume group activation means only one node will have write access to the volume group.

There are two requirements for exclusive activation:

- The volume group must be marked as a clustered volume group. The cluster ID on the volume group must match the cluster ID stored in the `/etc/cmcluster/cmclconfig` file. If the volume group has not been marked, then exclusive activation is not allowed.

The `cmlvmd` daemon process must be up and running. If `cmlvmd` is not running, there is no way to check if another node has already activated the volume group. The only way to start `cmlvmd` is to bring up the Serviceguard cluster. HP does not support running `cmlvmd` outside of Serviceguard.

## 6-5. SLIDE: Cluster Formation and Reformations

### Cluster Formation and Reformations



**Two concepts and an example of forming or reforming the cluster:**

- Ways to initially form the cluster
- Node failures, node exits, and node joins
- Cluster reformation example

### Student Notes

Another feature of Serviceguard clusters is the flexibility to allow nodes to easily exit and join existing clusters.

The next few slides highlight the cluster formation and reformation processes.

## 6–6. SLIDE: Ways to Initially Form the Cluster

### Ways to Initially Form the Cluster



**Two ways to initially form the cluster:**

**1. Manually**

Execute the `cmrunc1` command.

**2. Automatically at System Startup**

Edit the `/etc/rc.config.d/cmcluster` file.

```
AUTOSTART_CMCLD=1
```

## Student Notes

### Manual Cluster Startup

The `cmrunc1` command is used to manually initiate the cluster formation. Any node configured as part of the cluster can run this command if the cluster is not already up. This command is typically used the first time the cluster is brought up or after any cluster maintenance operation (or modification) that requires the cluster be brought down.

### Automatic Cluster Startup on Each Node

To enable each node in the cluster to automatically start or join the cluster at boot time, edit the `/etc/rc.config.d/cmcluster` file and set the value of `AUTOSTART_CMCLD` to 1.

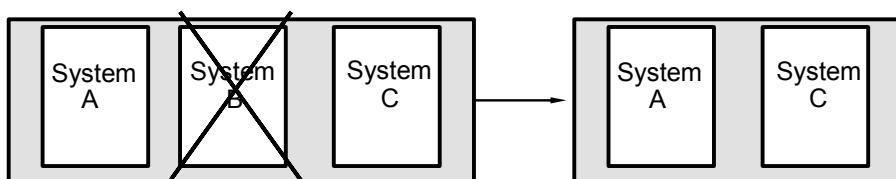
At boot time, if the cluster already exists, the node will try to join it; if no cluster exists, the node will attempt to start the cluster, provided all other nodes are available. If all other nodes are not yet available, the node will wait up to the period (as defined in the Cluster ASCII file) to form the cluster. If all nodes do not become available by the end of the `AUTO_START_TIMEOUT` period, the cluster will not form automatically.

## 6-7. SLIDE: Node Failures and Node Joins

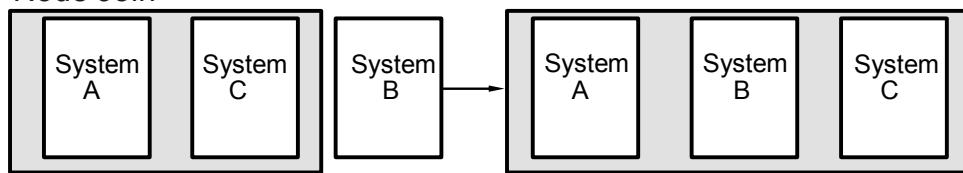
### Node Failures and Node Joins



Node Failure



Node Join



### Student Notes

The cluster reforms every time a change is made to cluster membership. The three methods by which membership changes are **node failure**, **node exit**, and **node join**.

#### Node Failure and Node Exit

Anytime a node leaves the cluster, a reformation takes place to exclude the exiting node. Nodes could leave the cluster for the following reasons:

- The `cshaltnode` command was run to manually remove the node from the cluster.
- A hardware failure occurred causing the node to crash.
- Heavy network traffic prohibits the heartbeat messages from being received causing the cluster coordinator to think the node was unhealthy.

## **Node Join**

Anytime an inactive node wants to join an existing cluster, a reformation takes place. Nodes join a cluster for the following reasons:

- The `cmrunnode` command was run to manually add the node to the cluster.
- A node reboots after a panic and is configured to automatically join the cluster at boot time.

## 6-8. SLIDE: Cluster Reformation Example

### Cluster Reformation Example

What happens when heartbeat packets cannot get through due to failure of a network component?

SystemA gets the cluster lock; SystemA reforms as a one-node cluster.

SystemB does not get cluster lock. Package2 moves to SystemA, and SystemB panics.

### Student Notes

The slide illustrates what happens during a cluster reformation in which both nodes receive exactly 50 percent of the vote.

### The Situation

Assume a two-node cluster, with Package1 running on SystemA and Package2 running on SystemB. Volume group vg01 is exclusively activated on SystemA, volume group vg02 is exclusively activated on SystemB. In addition, package IP addresses are assigned to SystemA and SystemB respectively (this is not shown on the slide).

### The Failure

Only one LAN has been configured for both heartbeat and data traffic. Failure of network equipment (such as a hub, switch, cable, or LAN card) would prevent heartbeat packets from getting through.

Since SystemA does not receive heartbeat messages from SystemB, SystemA attempts to reform as a one-node cluster. Likewise, since SystemB does not receive heartbeat messages from SystemA, SystemB also attempts to reform as a one-node cluster. During the election protocol, each node votes for itself, yielding both nodes with 50 percent of the vote. At this

point, a true race condition exists between SystemA and SystemB for the cluster lock disk. Because both nodes have 50 percent of the vote, both nodes go for the cluster lock. Only one will get it.

### **The Outcome**

Assume SystemA gets the cluster lock. SystemA reforms as a one-node cluster. After reformation, SystemA will make sure all applications configured to run on an existing clustered node are running. When SystemA discovers Package2 is not running in the cluster, it will try to start Package2 if Package2 is configured to run on SystemA.

SystemB recognizes that SystemA has begun reforming as a one node cluster because SystemA has the cluster lock. To release all resources related to Package2 ( such as exclusive access to volume group vg02 and the Package2 IP address ) as quickly as possible, SystemB will perform a transfer of control ( TOC ) panic.

## 6-9. SLIDE: Local LAN Card Failover

### Local LAN Card Failover



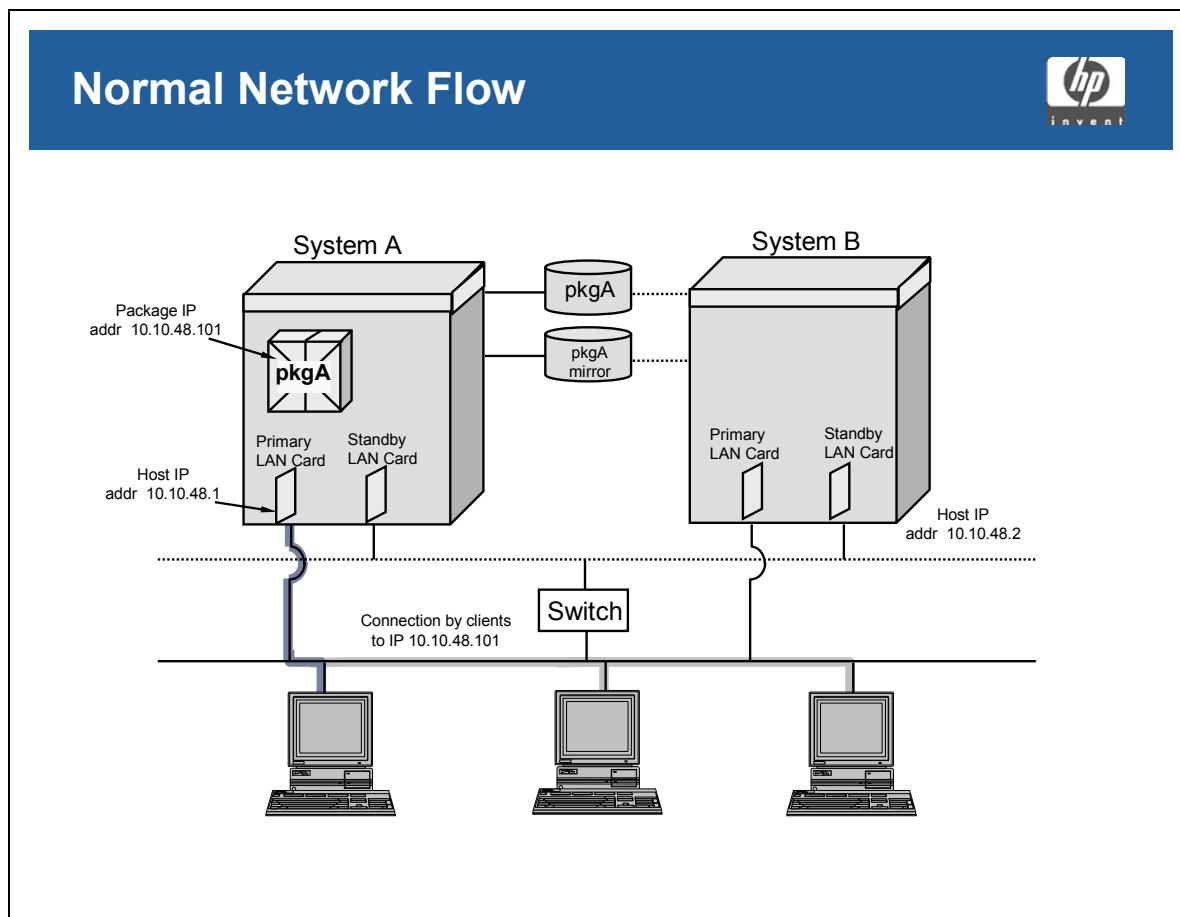
- Normal network flow through primary LAN card
- Network flow through standby LAN card

### Student Notes

The "local LAN card failover" feature allows every LAN card to have a corresponding standby LAN card. If the LAN card fails, all IP addresses on that LAN card would move to the standby LAN card, preventing an application outage from occurring.

The details of the local LAN card failover feature are illustrated on the next two slides.

## 6-10. SLIDE: Normal Network Flow

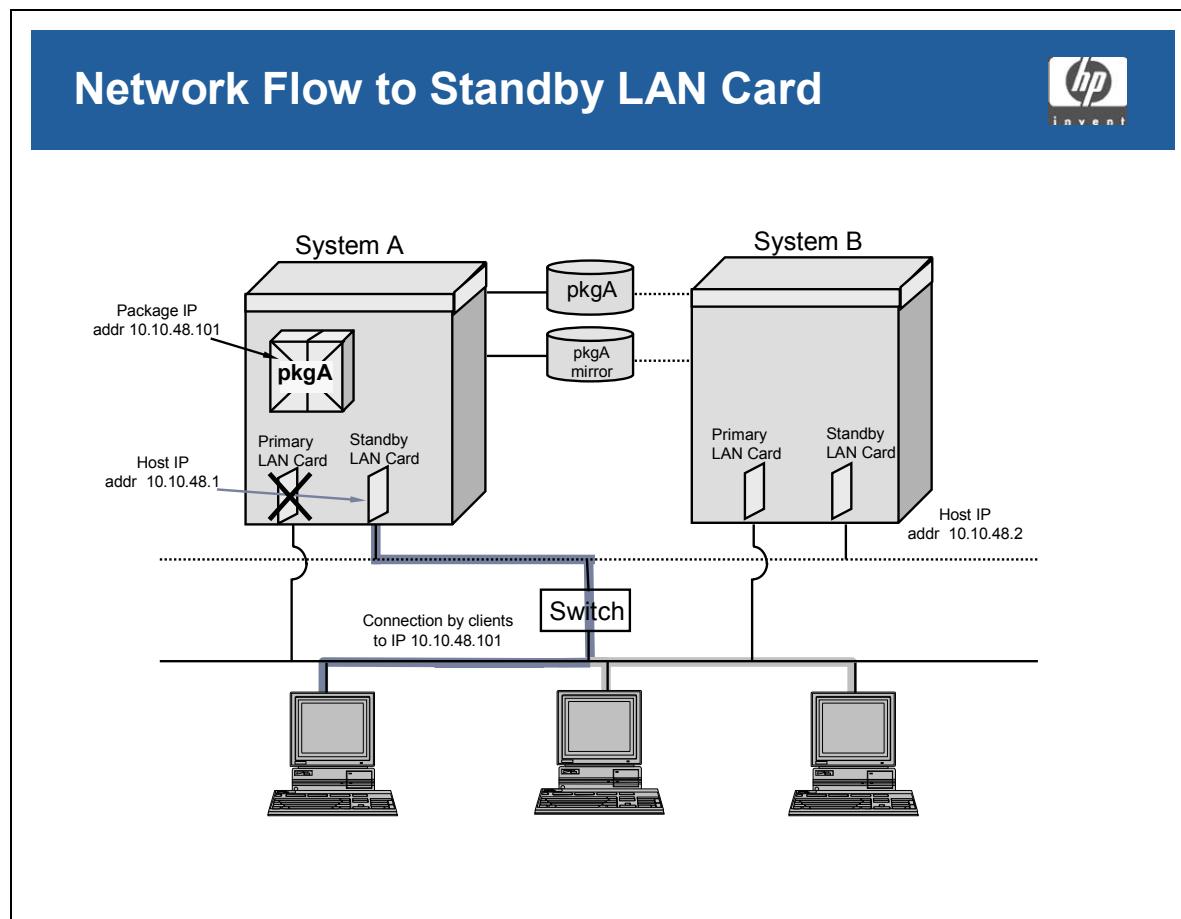


### Student Notes

The above slide shows two Serviceguard nodes connected to a single "bridged network", where the bridged network consists of two LAN segments ( an upper and lower segment ) joined by a hub. SystemA and SystemB are communicating with each other and the client systems over the lower LAN segment. The upper LAN segment in the slide is considered the "standby" segment.

Before turning to the next page, try to answer the following question: *"What would happen if the primary LAN card in SystemA were to fail?"*

## 6-11. SLIDE: Network Flow to Standby LAN Card



### Student Notes

In the above slide, we see the answer to the question: "What would happen if the primary LAN card in SystemA were to fail?"

When the primary LAN card fails, the Serviceguard daemon, `cmclsd`, detects the failure within the "NETWORK\_POLLING\_INTERVAL", and moves all IP addresses configured on the primary LAN card over to the standby LAN card. The switch is transparent at the TCP/IP level and all applications on SystemA continue to run on SystemA.

## 6-12. LAB: Shared Disks/Shared Volume Groups

### Directions

1. First, verify that the file system (/sgdata1) is not mounted on either node.  
(umount /sgdata1 if necessary)
2. Verify that vg01 is not activated on either node.
3. Ensure the cluster is up with both nodes as members.
4. Ensure that vg01 is marked as a "**cluster-aware**" volume group.
5. Try the following on node1:

```
# vgchange -a y vg01
```

What happens?

Module 6  
**Additional Cluster Features**

6. Again, test: on node1, issue the following command:

```
# vgchange -a e vg01
```

What happens?

7. Now test node2. On node2, issue the following command:

```
# vgchange -a e vg01
```

What happens?

8. Back on node1, issue the following command:

```
# vgchange -a n vg01
```

9. Now, bring down the cluster with a cmhaltcl command.

```
# cmhaltcl -v
```

10. With the cluster in a halted state, on node1, issue the following command:

```
# vgchange -a y vg01
```

This still does not work.

11. Now, on node1, issue the following command:

```
# vgchange -a e vg01
```

What happens?

12. Can we access our shared data in any way with the cluster down and the daemons not running? For example, we might want to run a database conversion or maintenance program with the cluster down. Can we do this?

13. On node1, issue the following commands:

```
# vgchange -c n vg01
# vgchange -a y vg01
# mount -F vxfs /dev/vg01/sglvol1 /sgdata1
```

14. Try accessing the data on the file system; use the traditional `ll` command. After that, unmount the filesystem.

15. Deactivate volume group `vg01`.

---

**NOTE:** You must return `vg01` to the cluster with `vgchange -c y vg01`. This, however, will not work without the Serviceguard daemons `cmlvmd` and `cmclld` running.

---

Module 6  
**Additional Cluster Features**

16. Mark the volume group `vg01` for cluster use. Does this command work properly?

17. Finally, restart the cluster.

## 6-13. LAB: Exiting and Joining the Cluster

### Directions

1. With the cluster up, both nodes as members, and  
`tail -f /var/adm/syslog/syslog.log` running on both systems, on `node1`, issue the `cmhaltnode` command.

```
# cmhaltnode           ← on <node1>
```

2. Check the following: Are both nodes still part of the cluster?

3. On `node1`, issue a `cmrunnode` command.

4. Next, configure **both** nodes to automatically join the cluster at boot time.

Module 6  
**Additional Cluster Features**

5. Now, shutdown and power off one of the nodes. Let us choose node2 for this purpose. Remember: if you are using local equipment, just physically turn-off the power switch. If you are using remote equipment, the shutdown command will automatically power off the system.
6. On the node that is still up, view the status of the cluster.
7. Next, halt the cluster.
8. With the cluster halted, attempt to start the cluster while one node is down.
9. Did the cluster start? Why or why not?
10. Force the cluster to start without the failed node. You will receive a warning. Reply "y" to continue to bring up the cluster.

11. Power-on node2, and allow it to come to run level 3. On remote equipment, go to the GSP / MP command menu and issue a pc command.
  12. Did node2 successfully rejoin the cluster? Briefly, please discuss why or why not.

## 6-14. LAB: Local LAN Card Failover (Using Remote Equipment)

### Special Note

- a. Lab 6-14 is for those using remote equipment (and do not have the ability to physically disconnect a LAN cable).
- b. If you have local equipment (and the ability to physically disconnect a LAN cable), do not do this lab. Instead, Turn to Lab 6-15.
- c. If you are not sure which equipment you have, ask your instructor.

**For this lab, open a Graphical (ReflectionX) session to each of your cluster's nodes and open an ASCII (Putty) session to <node1>.**

1. Execute the commands from the indicated nodes below. From `node2`, ping `node1`. Display the arp cache on `node2`. Make a note of the Ethernet and IP addresses of the LAN cards on `Node1` using `lanscan` and `netstat -in`.

**Answer:**

```
<node2># ping <node1>
<node2># arp -a | grep <node1>
```

Node1's IP Address: \_\_\_\_\_ Node1's MAC Address: \_\_\_\_\_

```
<node1># lanscan ; netstat -in
```

LAN IP \_\_\_\_\_

LAN MAC Address \_\_\_\_\_

LAN IP \_\_\_\_\_

LAN MAC Address \_\_\_\_\_

LAN IP \_\_\_\_\_

LAN MAC Address \_\_\_\_\_

2. On `node1`, and with the cluster up and with both nodes in the cluster, and in the Putty window, execute `tail -f /var/adm/syslog/syslog.log`. Back in `node1`'s Graphical window, issue the following:

```
<node1># xclock -update 1 -display ____(see comment below)__ &
```

To fill in the correct value in the blank space in the above command, use your ReflectionX window on `node2` and capture the output of the “`echo $DISPLAY`” command. Place this output in the blank in the above command. Also, do not forget to place this `xclock` process in the background.

3. While still on node1, you will run a script that simulates a failure of a LAN card. Watch the second hand on our xclock as you execute the following script:  
`/labs/mod_06/simulate_lan_card_failure.sh`. Allow this script to continue to run as you continue with step 4.

**Answer:**

```
<node1># /labs/mod_06/simulate_lan_card_failure.sh
```

4. What happens to the second hand on our xclock? What do you notice in the `syslog.log` file?

**Answer:**

The second hand pauses (reflecting the lan card failover), then skips forward again to reflect the correct time.

Additionally, `/var/adm/syslog/syslog.log` reflects the lan card failover.

5. In another window, view the cluster information.

6. Did Serviceguard detect the failure?

7. Did it automatically switch to the standby LAN card?

Module 6  
**Additional Cluster Features**

8. Issue an "arp node1" from the command line on node2 and note the IP address or Ethernet address of node1.

```
<node2># arp node1
```

9. Verify the LAN failover. Issue the commands lanscan and netstat -in on node1. Also, try another cmviewcl -v. This confirms that the system uses the backup LAN card when the primary LAN is down.

Compare the output with what you found in step #1 above.

10. Terminate the "simulate\_lan\_card\_failure.sh" with a <ctrl-c>. Then, using the commands from step 9 above, verify that Serviceguard automatically fails back to the primary LAN card. Note that it may take 10-15 seconds for the "failback" to complete.
11. Finally, identify the xclock process ID on node1. Use the PID to kill the process.

## 6-15. LAB: Local LAN Card Failover (Using Local Equipment)

### Directions

#### Special Note

- a. Lab 6-15 is for those using local equipment where you have the ability to physically disconnect a LAN cable.
- b. If you are using remote equipment, please do lab 6-14.
- c. If you are not sure which equipment you have, ask your instructor.

1. Execute the commands on from the indicated nodes below. From node2 , ping node1. Display the arp cache on node2. Make a note of the Ethernet and IP addresses of the LAN cards on Node1 using lanscan and netstat -in.

#### Answer:

```
<node2># ping <node1>
<node2># arp -a | grep <node1>
```

Node1's IP Address: \_\_\_\_\_ Node1's MAC Address: \_\_\_\_\_

```
<node1># lanscan ; netstat -in
```

LAN IP \_\_\_\_\_

LAN MAC Address \_\_\_\_\_

LAN IP \_\_\_\_\_

LAN MAC Address \_\_\_\_\_

LAN IP \_\_\_\_\_

LAN MAC Address \_\_\_\_\_

2. On node1 , and with the cluster up and with both nodes in the cluster, and in the Putty window, execute tail -f /var/adm/syslog/syslog.log . Back in node1's Graphical window, issue the following:

```
<node1># xclock -update 1 -display (see comment below) &
```

To fill in the correct value in the blank space in the above command, use your ReflectionX window on node2 and capture the output of the “echo \$DISPLAY” command. Place this output in the blank in the above command. Also, do not forget to place this xclock process in the background.

Module 6  
**Additional Cluster Features**

3. With the second hand moving on our `xclock` process, disconnect the primary LAN card on `node1` from the network.
  
  
  
  
  
  
4. What happens to the second hand on our `xclock`?
  
  
  
  
  
  
5. View the cluster information.
  
  
  
  
  
  
6. Did Serviceguard detect the failure?
  
  
  
  
  
  
7. Did it automatically switch to the standby LAN card?
  
  
  
  
  
  
8. Issue an “`arp node1`” from the command line on `node2` and note the IP address or Ethernet address of `node1`.

```
# arp node1
```

9. Verify the LAN failover. Issue the commands `lanscan` and `netstat -in` and `ifconfig lan0` and `ifconfig lan1` on `node1`. Also, try another `cmviewcl -v`. This confirms that the system uses the backup LAN card when the primary is disconnected. Compare the output with what you found in step #1 above.
  10. Reconnect the primary LAN card on `node1` and, using the commands from step 9 above, verify that Serviceguard automatically fails back to the primary LAN card.

Finally, identify the `xclock` process ID. Use the PID to kill the process.

## **6-16. LAB Solution: Shared Disks/Shared Volume Groups**

### **Directions**

1. First, verify that the file system (/sgdata1) is not mounted on either node.  
(umount /sgdata1 if necessary)

**Answer:**

```
# bdf  
# umount /sgdata1      ← Only if needed.
```

2. Verify that vg01 is not activated on either node.

**Answer:**

```
# vgdisplay vg01          ← on Node1 and on Node2  
# vgchange -a n vg01     ← if needed on either / both nodes
```

3. Ensure the cluster is up with both nodes as members.

**Answer:**

```
# cmviewcl
```

4. Ensure that vg01 is marked as a "**cluster-aware**" volume group.

**Answer:**

```
# vgchange -c y vg01
```

5. Try the following on node1:

```
# vgchange -a y vg01
```

What happens?

**Answer:**

The activation fails: "Conflicts with configured node."

6. Again, test: on `node1`, issue the following command:

```
# vgchange -a e vg01
```

What happens?

**Answer:**

Succeeds, since we used the "exclusive" option.

7. Now test `node2`. On `node2`, issue the following command:

```
# vgchange -a e vg01
```

What happens?

**Answer:**

Fails, since `node1` has the exclusive activation on `vg01`.

8. Back on `node1`, issue the following command:

```
# vgchange -a n vg01
```

9. Now, bring down the cluster with a `cmhaltcl` command.

```
# cmhaltcl -v
```

10. With the cluster in a halted state, on `node1`, issue the following command:

```
# vgchange -a y vg01
```

This still does not work.

Module 6  
**Additional Cluster Features**

11. Now, on node1, issue the following command:

```
# vgchange -a e vg01
```

What happens?

**Answer:**

Fails because the cluster is not running.

12. Can we access our shared data in any way with the cluster down and the daemons not running? For example, we might want to run a database conversion or maintenance program with the cluster down. Can we do this?

**Answer:**

Yes.

13. On node1, issue the following commands:

```
# vgchange -c n vg01
# vgchange -a y vg01
# mount -F vxfs /dev/vg01/sglvol1 /sgdata1
```

14. Try accessing the data on the file system; use the traditional `ll` command. After that, unmount the filesystem.

**Answer:**

```
# ll /sgdata1
# umount /sgdata1
```

15. Deactivate volume group `vg01`.

**Answer:**

```
# vgchange -a n vg01
```

---

**NOTE:** You must return `vg01` to the cluster with `vgchange -c y vg01`. This, however, will not work without the Serviceguard daemons `cmlvmd` and `cmclld` running.

---

16. Mark the volume group `vg01` for cluster use. Does this command work properly?

**Answer:**

```
# vgchange -c y vg01
```

No.

17. Finally, restart the cluster.

**Answer:**

```
# cmrunc1 -v
```

## 6-17. LAB Solution: Exiting and Joining the Cluster

### Directions

1. With the cluster up, both nodes as members, and `tail -f /var/adm/syslog/syslog.log` running on both systems, on `node1`, issue the `cmhaltnode` command.

```
# cmhaltnode           ← on <node1>
```

2. Check the following: Are both nodes still part of the cluster?

**Answer:**

```
# cmviewcl
No, <node1> is no longer in the cluster.
```

3. On `node1`, issue a `cmrunnode` command.

**Answer:**

```
# cmrunnode           ← on <node1>
```

4. Next, configure **both** nodes to automatically join the cluster at boot time.

**Answer:**

```
# cd /etc/rc.config.d
# vi cmcluster           ← On both nodes !
```

```
AUTOSTART_CMCLD=1
```

5. Now, shutdown and power off one of the nodes. Let us choose node2 for this purpose. Remember: if you are using local equipment, just physically turn-off the power switch. If you are using remote equipment, the shutdown command will automatically power off the system.

**Answer:**

```
# cd /
# shutdown -hy 0           ← On node2
<power-off>             ← Only with local equipment
```

6. On the node that is still up, view the status of the cluster.

**Answer:**

```
# cmviewcl -v
```

7. Next, halt the cluster.

**Answer:**

```
# cmhaltcl -v
```

8. With the cluster halted, attempt to start the cluster while one node is down.

**Answer:**

```
# cmrunc1 -v
```

9. Did the cluster start? Why or why not?

**Answer:**

No, since we have less than 100% of the nodes present.

10. Force the cluster to start without the failed node. You will receive a warning. Reply "y" to continue to bring up the cluster.

**Answer:**

```
<node1> # cmrunc1 -n <node1>
```

Module 6  
**Additional Cluster Features**

11. Power-on node2 , and allow it to come to run level 3. On remote equipment, go to the GSP / MP command menu and issue a pc command.

**Answer:**

<power-on> ← Using local equipment.

MP> cm ← Using remote equipment.  
MP:CM> pc

12. Did node2 successfully rejoin the cluster? Briefly, please discuss why or why not.

**Answer:**

<Node2> rejoins the cluster at boot because we set AUTOSTART\_CMCLD=1 in its /etc/rc.config.d/cmcluster file.

## 6-18. LAB Solution: Local LAN Card Failover (Using Remote Equipment)

### Special Note

- d. Lab 6-14 is for those using remote equipment (and do not have the ability to physically disconnect a LAN cable).
- e. If you have local equipment (and the ability to physically disconnect a LAN cable), do not do this lab. Instead, Turn to Lab 6-15.
- f. If you are not sure which equipment you have, ask your instructor.

**For this lab, open a Graphical (ReflectionX) session to each of your cluster's nodes and open an ASCII (Putty) session to <node1>.**

1. Execute the commands from the indicated nodes below. From node2, ping node1. Display the arp cache on node2. Make a note of the Ethernet and IP addresses of the LAN cards on Node1 using lanscan and netstat -in.

**Answer:**

```
<node2># ping <node1>
<node2># arp -a | grep <node1>
```

Node1's IP Address: \_\_\_\_\_ Node1's MAC Address: \_\_\_\_\_

```
<node1># lanscan ; netstat -in
```

LAN IP _____	LAN MAC Address _____
LAN IP _____	LAN MAC Address _____
LAN IP _____	LAN MAC Address _____

2. On node1, and with the cluster up and with both nodes in the cluster, and in the Putty window, execute tail -f /var/adm/syslog/syslog.log. Back in node1's Graphical window, issue the following:

```
<node1># xclock -update 1 -display (see comment below) &
```

To fill in the correct value in the blank space in the above command, use your ReflectionX window on node2 and capture the output of the "echo \$DISPLAY" command. Place this output in the blank in the above command. Also, do not forget to place this xclock process in the background.

Module 6  
**Additional Cluster Features**

3. While still on `node1`, you will run a script that simulates a failure of a LAN card. Watch the second hand on our `xclock` as you execute the following script:  
`/labs/mod_06/simulate_lan_card_failure.sh`. Allow this script to continue to run as you continue with step 4.

**Answer:**

```
<node1># /labs/mod_06/simulate_lan_card_failure.sh
```

4. What happens to the second hand on our `xclock`? What do you notice in the `syslog.log` file?

**Answer:**

The second hand pauses (reflecting the lan card failover), then skips forward again to reflect the correct time.

Additionally, `/var/adm/syslog/syslog.log` reflects the lan card failover.

5. In another window, view the cluster information.

**Answer:**

```
# cmviewcl -v
```

6. Did Serviceguard detect the failure?

**Answer:**

Yes. See entries in the `syslog.log` file and note that the primary lan card for `node1` shows a status of “down” in the `cmviewcl -v` output.

7. Did it automatically switch to the standby LAN card?

**Answer:**

Yes. See entries in the `syslog.log` file.

8. Issue an “arp node1” from the command line on node2 and note the IP address or Ethernet address of node1.

```
<node2># arp node1
```

9. Verify the LAN failover. Issue the commands lanscan and netstat -in on node1. Also, try another cmviewcl -v. This confirms that the system uses the backup LAN card when the primary LAN is down.

**Answer:**

```
# lanscan
# netstat -in
# cmviewcl -v
```

Compare the output with what you found in step #1 above.

10. Terminate the “simulate\_lan\_card\_failure.sh” with a <ctrl-c>. Then, using the commands from step 9 above, verify that Serviceguard automatically fails back to the primary LAN card. Note that it may take 10-15 seconds for the “failback” to complete.

**Answer:**

```
< ctrl-c >
# lanscan
# netstat -in
# ifconfig lan0
# cmviewcl -v
```

11. Finally, identify the xclock process ID on node1. Use the PID to kill the process.

**Answer:**

```
# ps -ef | grep xclock
# kill <pid>
```

---

## 6-19. LAB Solution: Local LAN Card Failover (Using Local Equipment)

### Directions

#### Special Note

- d. Lab 6-15 is for those using local equipment where you have the ability to physically disconnect a LAN cable.
- e. If you are using remote equipment, please do lab 6-14.
- f. If you are not sure which equipment you have, ask your instructor.

1. Execute the commands on from the indicated nodes below. From node2 , ping node1. Display the arp cache on node2. Make a note of the Ethernet and IP addresses of the LAN cards on Node1 using lanscan and netstat -in.

#### Answer:

```
<node2># ping <node1>
<node2># arp -a | grep <node1>
```

Node1's IP Address: \_\_\_\_\_ Node1's MAC Address: \_\_\_\_\_

```
<node1># lanscan ; netstat -in
```

LAN IP _____	LAN MAC Address _____
LAN IP _____	LAN MAC Address _____
LAN IP _____	LAN MAC Address _____

2. On node1 , and with the cluster up and with both nodes in the cluster, and in the Putty window, execute tail -f /var/adm/syslog/syslog.log . Back in node1's Graphical window, issue the following:

```
<node1># xclock -update 1 -display ____(see comment below)____ &
```

To fill in the correct value in the blank space in the above command, use your ReflectionX window on node2 and capture the output of the “echo \$DISPLAY” command. Place this output in the blank in the above command. Also, do not forget to place this xclock process in the background.

3. With the second hand moving on our `xclock` process, disconnect the primary LAN card on `node1` from the network.

**Answer:**

Unplug the LAN cable connected to the primary LAN card.

4. What happens to the second hand on our `xclock`?

**Answer:**

The second hand pauses (reflecting the lan card failover), then skips forward again to reflect the correct time.

Additionally, `/var/adm/syslog/syslog.log` reflects the lan card failover.

5. View the cluster information.

**Answer:**

```
# cmviewcl -v
```

6. Did Serviceguard detect the failure?

**Answer:**

Yes. See the `syslog.log` file.

7. Did it automatically switch to the standby LAN card?

**Answer:**

Yes. See the `syslog.log` file.

8. Issue an “`arp node1`” from the command line on `node2` and note the IP address or Ethernet address of `node1`.

```
# arp node1
```

9. Verify the LAN failover. Issue the commands `lanscan` and `netstat -in` and `ifconfig lan0` and `ifconfig lan1` on `node1`. Also, try another `cmviewcl -v`. This confirms that the system uses the backup LAN card when the primary is disconnected.

**Answer:**

```
# lanscan
# netstat -in
# ifconfig lan0
# cmviewcl -v
```

Compare the output with what you found in step #1 above.

10. Reconnect the primary LAN card on `node1` and, using the commands from step 9 above, verify that Serviceguard automatically fails back to the primary LAN card.

**Answer:**

```
# lanscan
# netstat -in
# ifconfig lan0
# cmviewcl -v
```

11. Finally, identify the `xclock` process ID. Use the PID to kill the process.

**Answer:**

```
# ps -ef | grep xclock
# kill <pid>
```

---

# **Module 7 — Packages and Services**

## **Objectives**

Upon completion of this module, you will be able to do the following:

- Configure a basic package to run in a Serviceguard environment.
- Describe the fields in a package configuration file.
- Describe the variables in a package control script file.
- Provide the command to re-enable a package for switching.
- Interpret the output of `cmviewcl -v` as it relates to packages.
- Provide the full path name of a package-specific log file.

## 7-1. SLIDE: Packaging Concepts

### Packaging Concepts



- A package is an application and all the application's resources required to execute properly.
- Resources for packages may include:
  - Disk Resources ( Volume Group(s), Disk Group(s) )
  - Network Resources ( IP Address(es) )
  - Service Processes ( At least one process is required )
- Packaging Files include:
  - Package Configuration File ( pkg.conf )
  - Package Control Script ( pkg.cntl )

### Student Notes

Applications that run in a Serviceguard high availability environment must be configured with all their related resources into a **package**.

The information needed to run a package in a Serviceguard cluster is contained in the ...

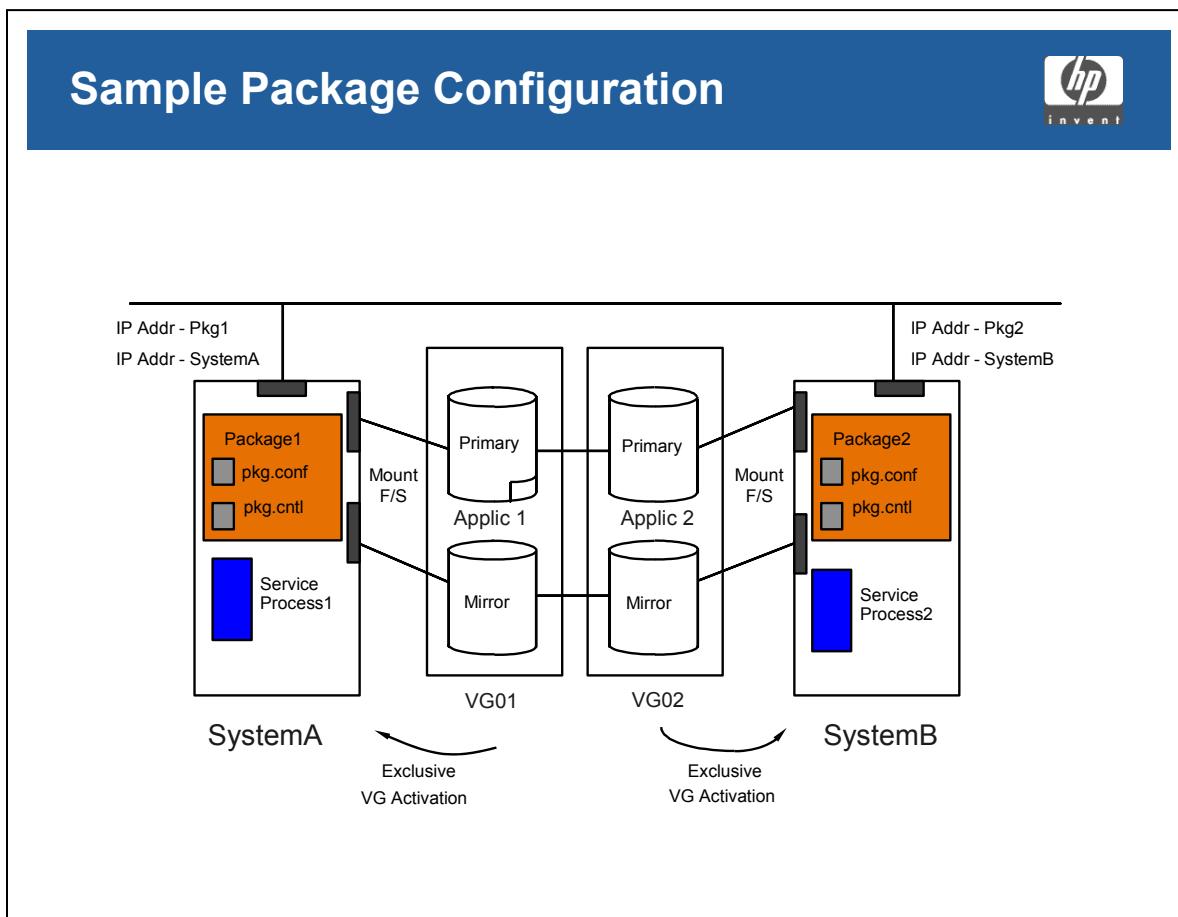
- **package configuration file** and
- **package control script**.

One configuration file and one control script will need to exist for each package.

The package configuration file defines the dependencies for the application, like a subnet or a service process. Also defined in the configuration file are package attributes and characteristics.

The package control script is executed to bring up ( or bring down ) the packaged application in a Serviceguard environment.

## 7-2. SLIDE: Sample Package Configuration



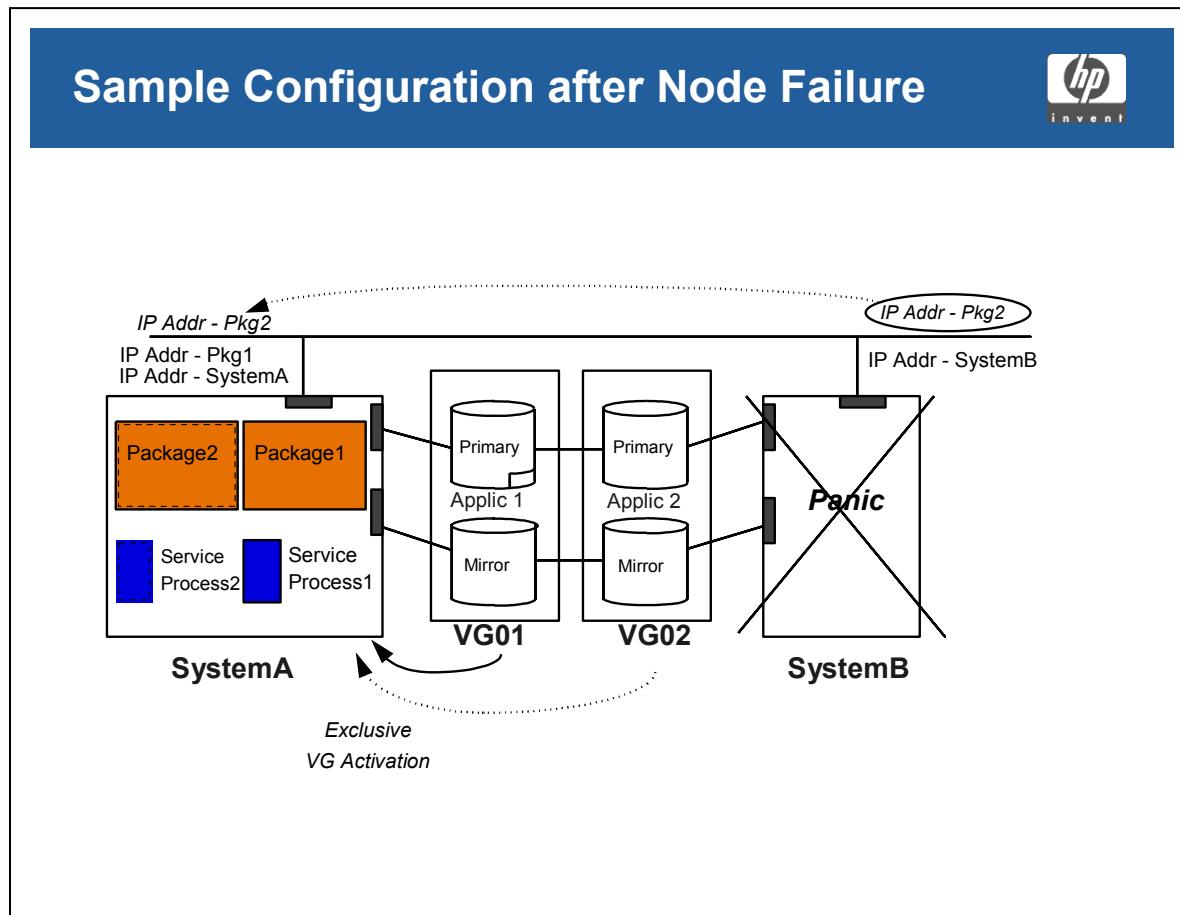
### Student Notes

The slide shows a sample Serviceguard cluster configured with two packages.

Package1 was started on SystemA using the `pkg.conf` and the `pkg.cntl` files. Three things should be noted related to the execution of Package1 on SystemA:

- SystemA has volume group `vg01` exclusively activated. This is because `vg01` contains the application specific data for Package1.
- The LAN card on SystemA contains two IP addresses: one for the host SystemA and one for Package1.
- The service process for Package1 is currently running on SystemA. Should the service process terminate, Serviceguard will interpret that as an application failure and will try to restart the service process or move the application over to SystemB.

### 7-3. SLIDE: Sample Configuration after Node Failure



### Student Notes

The above diagram illustrates what happens when SystemB fails (assume a CPU or memory board failure).

At NODE\_TIMEOUT seconds, SystemA will detect that SystemB is no longer sending heartbeat messages. SystemA will respond by attempting to reform the cluster without SystemB. Since SystemA makes up exactly 50 percent of the cluster, it will need to get the cluster lock disk. This will not be a problem because SystemB is down and will not be competing for the cluster lock disk.

Once the cluster reforms, SystemA detects Package2 is not running in the cluster. If Package2 is configured to run on SystemA, SystemA runs the package script for Package2, which exclusively activates vg02 onto SystemA, assigns the Package2's IP address to the LAN card on SystemA, and starts the service process for Package2 on SystemA. SystemA is now running both packages.

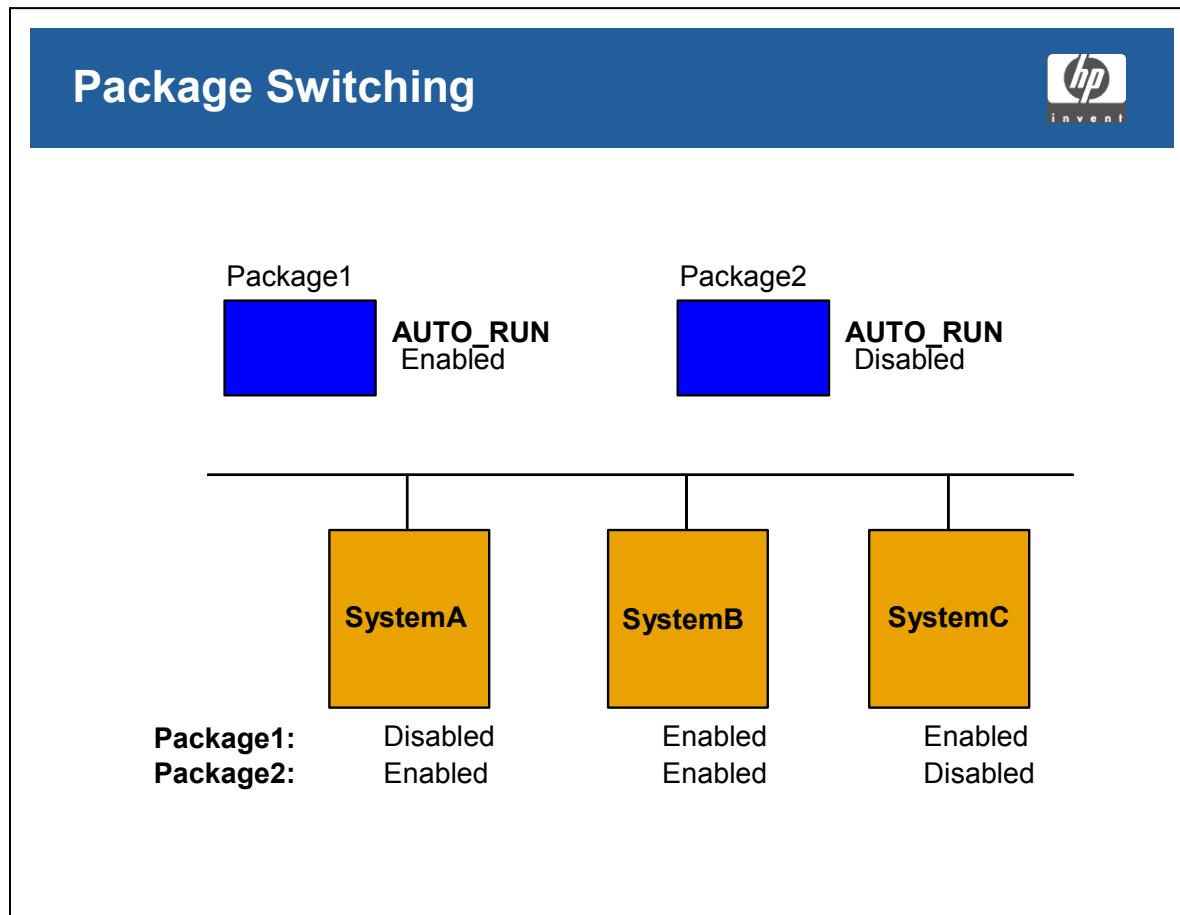
Once SystemB is fixed, it can be rebooted and the cluster will reform to be a 2-node cluster again with SystemB. However, Package2 will *not* automatically move back to SystemB during reformation. Moving the package back to SystemB would require the application be brought down (hence an application outage). Unless Package2 is configured to move back to SystemB

automatically, Serviceguard leaves it on SystemA. Package behavior is configurable on a per-package basis.

Package2 could be manually moved back by the system administrator by executing the following commands:

```
# cmhaltpkg -v Package2  
and then  
# cmrunkpkg -v -n SystemB Package2
```

## 7-4. SLIDE: Package Switching



### Student Notes

Two parameters need to be enabled for a package to switch between nodes in a cluster:  
`AUTO_RUN` and `NODE_SWITCHING`.

#### Auto Run

Every package has an attribute called **AUTO\_RUN**, which determines if a package can be run automatically among nodes in the cluster. If `AUTO_RUN` is disabled, the package will not move to other nodes, even if the node was enabled to receive the package.

#### Node Switching

Every node has an attribute per package called **SWITCHING**, which determines if that package can be received (switched to) by that node. If `SWITCHING` is disabled on a node for a particular package, that package will not move to that particular node, even though the package is enabled for package switching.

#### Example

There are two packages shown on the slide: Package1 and Package2. Package1 is capable of switching to other nodes within the cluster, but Package2 is not capable of switching.

The three nodes shown on the slide are selectively enabled / disabled for the various packages. Because Package2 is disabled, the fact that SystemB and SystemC can receive the package is irrelevant. Also, note that Package1 will not move to SystemA due to SystemA being disabled for Package1.

The only package movement that can take place is Package1 moving to SystemB or SystemC.

## 7-5. SLIDE: Viewing Package Status

### Viewing Package Status

hp invent

```
# cmviewcl -v

CLUSTER      STATUS
cluster1     up

NODE          STATUS      STATE
SystemA       up          running

Network_Parameters:
INTERFACE      STATUS      PATH      NAME
PRIMARY        up          0/1/2/0   lan1
STANDBY        up          0/2/1/0   lan2

PACKAGE        STATUS      STATE    AUTO_RUN      NODE
Package1       up          running   enabled      SystemA

Script_Parameters:
ITEM          STATUS      NAME      MAX_RESTARTS  RESTARTS
Service        up          servicel.1 2           0
Subnet         up          10.10     0           0

Node_Switching_Parameters:
NODE_TYPE      STATUS      SWITCHING      NAME
Primary        up          enabled        SystemA
Alternate      up          disabled       SystemB
```

### Student Notes

The status of packages and services can be seen with the `cmviewcl` command. Package related fields to monitor in this output include the following:

- **AUTO\_RUN** specifies whether a package can switch to other nodes should the package fail.
- **SWITCHING** specifies which nodes on a per package basis are able to receive the package.
- **Script\_parameters** are resources in which the package is dependent upon in order to be considered healthy. These resources are configured in the package's run script.
- The above screen shot shows a cluster configured with LVM disks providing cluster lock services. Had this cluster been configured using a Quorum Server, then the Quorum Server would also be seen if the -v ( verbose ) option was used with the `cmviewcl` command.

## 7-6. SLIDE: Modifying Package Status

### Modifying Package Status



#### Enable Package for Switching To Any Node:

```
cmmodpkg -e pkg_name
```

#### Enable a Specific Node to Receive a Package:

```
cmmodpkg -e -n nodename pkg_name
```

### Student Notes

The `cmmodpkg` command performs two operations. It enables / disables a package's ability to switch to another node, and it enables / disables a particular node from running specific packages.

The slide shows the syntax for enabling ( `-e` option ) a package or node for switching. The Serviceguard Administrator also would also commonly use the `-d` option to disable a package or node for switching.

Therefore, the complete syntax for the `cmmodpkg` command actually takes two forms:

- **To enable or disable a package for switching:**

```
# cmmodpkg [ -v ] { -e | -d } pkg_name
```

- **To enable or disable a node to / from receiving a package :**

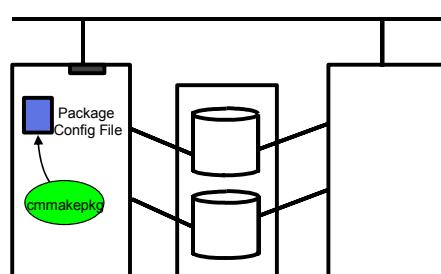
```
# cmmodpkg [ -v ] { -e | -d } -n node_name pkg_name
```

## 7-7. SLIDE: Step 1: Build Package Configuration File

### Step 1: Build Package Configuration File



```
PACKAGE_NAME
FAILOVER_POLICY
FAILBACK_POLICY
NODE_NAME      original_node
NODE_NAME      adoptive_node
AUTO_RUN
NODE_FAIL_FAST_ENABLED
RUN_SCRIPT
RUN_SCRIPT_TIMEOUT
HALT_SCRIPT
HALT_SCRIPT_TIMEOUT
SERVICE_NAME
SERVICE_FAIL_FAST_ENABLED
SERVICE_HALT_TIMEOUT
SUBNET
RESOURCE_NAME
RESOURCE_POLLING_INTERVAL
RESOURCE_START
RESOURCE_UP_VALUE
```



The diagram illustrates the relationship between a package configuration file and its resources. A central box labeled "Package Config File" is connected to three separate boxes, each containing two cylinders. These cylinders represent resources, and they are all connected to a vertical line representing a node.

**Step-by-Step Procedure to Package an Application**

1. Build the Package Configuration File
2. Build the Package Run/Halt Script
3. Compile and Distribute the Binary File

### Student Notes

There are three main steps for packaging an application for a Serviceguard environment. The three main steps are:

1. Build the package configuration file.
2. Build the package run/halt script.
3. Compile and distribute the binary file.

### Step 1: Build the Package Configuration File

The first step in building the package configuration file is to run the `cmmakepkg` command to create the package configuration template:

```
# cmmakepkg -v -p package.conf
```

The next step is to edit / customize the file with the `vi` editor:

```
# vi package.conf
```

The following are some of the more important fields in the package configuration file.

<i>PACKAGE_NAME</i>	The name of the package. The package name must be unique to the cluster. <i>PACKAGE_NAME</i> can be as long as 39 characters.
<i>PACKAGE_TYPE</i>	This indicates whether this package is to run as a <b>FAILOVER</b> or <b>SYSTEM_MULTI_NODE</b> package.
<b>FAILOVER</b>	Package runs on one node at a time and if a failure occurs it can switch to an alternate node.
<b>SYSTEM_MULTI_NODE</b>	Package runs on multiple nodes at the same time. It cannot be started and halted on individual nodes. The only package that runs with this attribute is Cluster Volume Manager (CVM). Further, both <i>NODE_FAIL_FAST_ENABLED</i> and <i>AUTO_RUN</i> must be set to YES for this type of package. All <i>SERVICES</i> must have <i>SERVICE_FAIL_FAST_ENABLED</i> set to YES.
<i>FAILOVER_POLICY</i>	The failover policy determines how Serviceguard selects a failover node when the package fails.
<i>FAILBACK_POLICY</i>	The fallback policy determines whether Serviceguard returns the package back to its original node when the node is available again.
<i>NODE_NAME</i>	The names of the primary node and adoptive nodes for the package. The order is important. The first listed will be the primary node. The second listed will be the first adoptive. This would be followed by additional adoptive node names in the order of failover.
<i>AUTO_RUN</i>	The <i>AUTO_RUN</i> parameter determines whether the package will automatically be started when the cluster is started. <i>AUTO_RUN</i> replaces <i>PKG_SWITCHING_ENABLED</i> , which was obsoleted in MC/ServiceGuard, version 11.12.
<i>LOCAL_LAN_FAILOVER_ALLOWED</i>	The <i>LOCAL_LAN_FAILOVER_ALLOWED</i> parameter determines if the cluster software will be allowed to switch LANs locally ( i.e. transfer to a standby LAN card ) in the event of a primary LAN card failure.
<i>NODE_FAIL_FAST_ENABLED</i>	When set to YES ( NO is the default ), this parameter will cause the node to TOC panic if the package halt script fails or if the node loses contact with its subnet.

Module 7  
**Packages and Services**

<i>SCRIPT_LOG_FILE</i>	The default filename and location for the package logfile is /etc/cmcluster/pkgname/pkg_control_script.log. If you wish to rename this file and/or change the location, you may uncomment this line and specify an absolute path and filename to the alternate location.
<i>RUN_SCRIPT</i>	The absolute path name of the package run script. It is recommended this be the same file as the <i>HALT_SCRIPT</i> .
<i>RUN_SCRIPT_TIMEOUT</i>	The amount of time to allow the <i>RUN_SCRIPT</i> to complete. If the script does not complete within the time out value, then the package is halted and disabled.
<i>HALT_SCRIPT</i>	The full path name of the package halt script. It is recommended this be the same file as the <i>RUN_SCRIPT</i> .
<i>HALT_SCRIPT_TIMEOUT</i>	The amount of time, in seconds, to allow the <i>HALT_SCRIPT</i> to complete. If the script does not complete within the time out value, then the package is halted and disabled.
<i>STORAGE_GROUP</i>	Reserved for use with CVM.
<i>SERVICE_NAME</i>	The Serviceguard name of a service. A <i>SERVICE_NAME</i> should be unique, single words, without quotes. <i>SERVICE_NAME</i> is the link between the package configuration script and the package control ( run / halt ) script, and must be the same in both files.
<i>SERVICE_FAIL_FAST_ENABLED</i>	This parameter indicates whether the failure of the service should result in a TOC panic. If set to YES, a TOC panic will occur if this service fails. The default is NO.
<i>SERVICE_HALT_TIMEOUT</i>	The amount of time, in seconds, to allow the service process to be terminated with the <b>SIGTERM</b> signal ( -15 ). If the service does not terminate within the time out value, the <b>SIGKILL</b> signal ( -9 ) will be sent to the service process.
<i>SUBNET</i>	The health of the package depends on this subnet address. If Serviceguard can't access the subnet, the package will fail. With <i>NODE_FAILFAST_ENABLED</i> set to YES, the node will TOC panic. Regardless of whether or not the package will be assigned an IP address, the <i>SUBNET</i> line must have an entry.
<i>RESOURCE_NAME</i>	This line is the fully qualified resource path name.
<i>RESOURCE_POLLING_INTERVAL</i>	This is an optional field. If missing, the default value is 60, and indicates how often, in seconds, the resource is to be monitored.

*RESOURCE\_START\_VALUE*

This option can be either AUTOMATIC or DEFERRED. The default is AUTOMATIC, and means that Serviceguard will start resource monitoring automatically when the node starts up. If DEFERRED is selected, Serviceguard will not attempt to start up resource monitoring when the node starts up. Rather, the user is expected to specify all the DEFERRED resources in the package run script. In this way, the DEFERRED resources will be started up from the package run script during package run time.

*RESOURCE\_UP\_VALUE*

We must always have one or more of these RESOURCE\_UP\_VALUE lines. Specifics options are documented in the package configuration file itself.

Access Control Policy Parameters

By default, Serviceguard configuration can only be performed by the system administrator. Access control policies allow the root user to delegate certain tasks to non-root users. The package configuration file has entries for USER\_NAME, USER\_HOST, and USER\_ROLE. USER\_NAME is the user to whom certain access will be granted. USER\_HOST is where USER\_NAME can issue Serviceguard commands (not necessarily a node in the cluster). USER\_ROLE defines USER\_NAME's privileges and must be PACKAGE\_ADMIN.

---

## 7-8. TEXT PAGE: Sample Package Configuration File

### HIGH AVAILABILITY PACKAGE CONFIGURATION FILE (template)

This is an example of a configuration file.

---

**NOTE:** This file MUST be edited before it can be used. For complete details about package parameters and how to set them, consult the Serviceguard, Serviceguard OPS edition, or Serviceguard SGeRAC man pages or manuals.

---

#### Special Note

**The following package configuration file is from version 11.17 of Serviceguard.**

```
# ****
# ***** HIGH AVAILABILITY PACKAGE CONFIGURATION FILE (template) *****
# ****
# ***** Note: This file MUST be edited before it can be used. *****
# * For complete details about package parameters and how to set them, *
# * consult the Serviceguard Extension for RAC manuals.
# ****

# Enter a name for this package. This name will be used to identify the
# package when viewing or manipulating it. It must be different from
# the other configured package names.

PACKAGE_NAME

# Enter the package type for this package. PACKAGE_TYPE indicates
# whether this package is to run as a FAILOVER, MULTI_NODE, or
# SYSTEM_MULTI_NODE package.
#
#     FAILOVER      package runs on one node at a time and if a failure
#                   occurs it can switch to an alternate node.
#
#     MULTI_NODE    package runs on multiple nodes at the same time and
#                   can be independently started and halted on
#                   individual nodes. Failures of package components such
#                   as services, EMS resources or subnets, will cause
#                   the package to be halted only on the node on which the
#                   failure occurred. Relocatable IP addresses cannot be
#                   assigned to MULTI_NODE packages.
#
#     SYSTEM_MULTI_NODE
#                   package runs on all cluster nodes at the same time.
#                   It can not be started and halted on individual nodes.
#                   Both NODE_FAIL_FAST_ENABLED and AUTO_RUN must be set
#                   to YES for this type of package. All SERVICES must
#                   have SERVICE_FAIL_FAST_ENABLED set to YES.
#
# NOTE: Packages which have a PACKAGE_TYPE of MULTI_NODE and
#       SYSTEM_MULTI_NODE are not failover packages and are only
```

```
# supported for use by applications provided by Hewlett-Packard.  
  
# Since MULTI_NODE and SYSTEM_MULTI_NODE packages can run on more  
# than one node at a time and do not failover in the event of a  
# package failure, the following parameters cannot be  
# specified when configuring packages of these types:  
  
#     FAILOVER_POLICY  
#     FAILBACK_POLICY  
  
# Since an IP address can not be assigned to more than one node at a  
# time, relocatable IP addresses can not be assigned in the  
# package control script for MULTI_NODE packages. If volume  
# groups are used in a MULTI_NODE package, they must be  
# activated in a shared mode and data integrity is left to the  
# application. Shared access requires a shared volume manager.  
  
# Examples : PACKAGE_TYPE      FAILOVER (default)  
#             PACKAGE_TYPE      MULTI_NODE  
#             PACKAGE_TYPE      SYSTEM_MULTI_NODE
```

## FAILOVER

```
# Enter the failover policy for this package. This policy will be used
# to select an adoptive node whenever the package needs to be started.
# The default policy unless otherwise specified is CONFIGURED_NODE.
# This policy will select nodes in priority order from the list of
# NODE_NAME entries specified below.
#
# The alternative policy is MIN_PACKAGE_NODE. This policy will select
# the node, from the list of NODE_NAME entries below, which is
# running the least number of packages at the time this package needs
# to start.
```

## **FAILOVER POLICY**

## **CONFIGURED NODE**

```
# Enter the fallback policy for this package. This policy will be used  
# to determine what action to take when a package is not running on  
# its primary node and its primary node is capable of running the  
# package. The default policy unless otherwise specified is MANUAL.  
# The MANUAL policy means no attempt will be made to move the package  
# back to its primary node when it is running on an adoptive node.  
#  
# The alternative policy is AUTOMATIC. This policy will attempt to  
# move the package back to its primary node whenever the primary node  
# is capable of running the package.
```

## **FALLBACK POLICY**

## **MANUAL**

# Enter the names of the nodes configured for this package. Repeat  
# this line as necessary for additional client nodes.

```
# this line as necessary for additional adoptive nodes.  
#  
# NOTE: The order is relevant.  
#       Put the second Adoptive Node after the first one.  
#  
# Example : NODE_NAME    original_node  
#           NODE_NAME    adoptive_node
```

## Module 7 Packages and Services

```
# If all nodes in the cluster are to be specified and order is not
# important, "NODE_NAME *" may be specified.
#
# Example : NODE_NAME  *

NODE_NAME

# Enter the value for AUTO_RUN. Possible values are YES and NO.
# The default for AUTO_RUN is YES. When the cluster is started the
# package will be automatically started. In the event of a failure the
# package will be started on an adoptive node. Adjust as necessary.
#
# AUTO_RUN replaces obsolete PKG_SWITCHING_ENABLED.

AUTO_RUN          YES

# Enter the value for LOCAL_LAN_FAILOVER_ALLOWED.
# Possible values are YES and NO.
# The default for LOCAL_LAN_FAILOVER_ALLOWED is YES. In the event of a
# failure, this permits the cluster software to switch LANs locally
# (transfer to a standby LAN card). Adjust as necessary.
#
# LOCAL_LAN_FAILOVER_ALLOWED replaces obsolete NET_SWITCHING_ENABLED.

LOCAL_LAN_FAILOVER_ALLOWED    YES

# Enter the value for NODE_FAIL_FAST_ENABLED.
# Possible values are YES and NO.
# The default for NODE_FAIL_FAST_ENABLED is NO. If set to YES,
# in the event of a failure, the cluster software will halt the node
# on which the package is running. All SYSTEM_MULTI_NODE packages must have
# NODE_FAIL_FAST_ENABLED set to YES. Adjust as necessary.

NODE_FAIL_FAST_ENABLED        NO

# Enter the complete path for the run and halt scripts. In most cases
# the run script and halt script specified here will be the same script,
# the package control script generated by the cmmakepkg command. This
# control script handles the run(ning) and halt(ing) of the package.
#
# Enter the timeout, specified in seconds, for the run and halt scripts.
# If the script has not completed by the specified timeout value,
# it will be terminated. The default for each script timeout is
# NO_TIMEOUT. Adjust the timeouts as necessary to permit full
# execution of each script.
#
# Note: The HALT_SCRIPT_TIMEOUT should be greater than the sum of
# all SERVICE_HALT_TIMEOUT values specified for all services.
#
# The file where the output of the scripts is logged can be specified
# via the SCRIPT_LOG_FILE parameter. If not set, script output is sent
# to a file named by appending '.log' to the script path.
#
#SCRIPT_LOG_FILE

RUN_SCRIPT
RUN_SCRIPT_TIMEOUT           NO_TIMEOUT
HALT_SCRIPT
HALT_SCRIPT_TIMEOUT          NO_TIMEOUT
```

```

# Enter the names of the storage groups configured for this package.
# Repeat this line as necessary for additional storage groups.
#
# Storage groups are only used with CVM disk groups. Neither
# VxVM disk groups or LVM volume groups should be listed here.
# By specifying a CVM disk group with the STORAGE_GROUP keyword
# this package will not run until the CVM system multi node package is
# running and thus the CVM shared disk groups are ready for
# activation.
#
# NOTE: Should only be used by applications provided by
#       Hewlett-Packard.
#
# Example : STORAGE_GROUP    dg01
#           STORAGE_GROUP    dg02
#           STORAGE_GROUP    dg03
#           STORAGE_GROUP    dg04
#
# Enter the names of the dependency condition for this package.
# Dependencies are used to describe the relationship between packages
# To define a dependency, all three attributes are required.
#
# DEPENDENCY_NAME must have a unique identifier for the dependency.
#
# DEPENDENCY_CONDITION
#   This is an expression describing what must be true for
#   the dependency to be satisfied.
#
#   The syntax is: <package name> = UP , where <package name>
#   is the name of a multi-node or system multi-node package.
#
# DEPENDENCY_LOCATION
#   This describes where the condition must be satisfied.
#   The only possible value for this attribute is SAME_NODE
#
# NOTE:
# Dependencies can only be defined for packages within a CFS cluster. These are
# automatically setup in the SYSTEM-MULTI-NODE and MULTI-NODE packages created for
# disk groups and mount points. Customers configure dependencies for FAILOVER type
# packages only; and the dependency would be on a MULTI-NODE mount point (MP) package.
#
# Example :
#           DEPENDENCY_NAME      SG-CFS-MP-1
#           DEPENDENCY_CONDITION  SG-CFS-MP-1=UP
#           DEPENDENCY_LOCATION   SAME_NODE
#
#DEPENDENCY_NAME
#DEPENDENCY_CONDITION
#DEPENDENCY_LOCATION SAME_NODE
#
# Enter the SERVICE_NAME, the SERVICE_FAIL_FAST_ENABLED and the
# SERVICE_HALT_TIMEOUT values for this package. Repeat these
# three lines as necessary for additional service names. All
# service names MUST correspond to the SERVICE_NAME[] entries in
# the package control script.
#
# The value for SERVICE_FAIL_FAST_ENABLED can be either YES or
# NO. If set to YES, in the event of a service failure, the

```

## Module 7

### Packages and Services

```
# cluster software will halt the node on which the service is
# running. If SERVICE_FAIL_FAST_ENABLED is not specified, the
# default will be NO.
#
# SERVICE_HALT_TIMEOUT is represented as a number of seconds.
# This timeout is used to determine the length of time (in
# seconds) the cluster software will wait for the service to
# halt before a SIGKILL signal is sent to force the termination
# of the service. In the event of a service halt, the cluster
# software will first send a SIGTERM signal to terminate the
# service. If the service does not halt, after waiting for the
# specified SERVICE_HALT_TIMEOUT, the cluster software will send
# out the SIGKILL signal to the service to force its termination.
# This timeout value should be large enough to allow all cleanup
# processes associated with the service to complete. If the
# SERVICE_HALT_TIMEOUT is not specified, a zero timeout will be
# assumed, meaning the cluster software will not wait at all
# before sending the SIGKILL signal to halt the service.
#
# Example: SERVICE_NAME           DB_SERVICE
#           SERVICE_FAIL_FAST_ENABLED    NO
#           SERVICE_HALT_TIMEOUT       300
#
# To configure a service, uncomment the following lines and
# fill in the values for all of the keywords.
#
#SERVICE_NAME          <service name>
#SERVICE_FAIL_FAST_ENABLED <YES/NO>
#SERVICE_HALT_TIMEOUT   <number of seconds>

# Enter the network subnet name that is to be monitored for this package.
# Repeat this line as necessary for additional subnet names. If any of
# the subnets defined goes down, the package will be switched to another
# node that is configured for this package and has all the defined subnets
# available.
# The subnet names could be IPv4 or IPv6. The network subnet
# names that are to be monitored for this package could be a mix
# of IPv4 or IPv6 subnet names

#SUBNET

# The keywords RESOURCE_NAME, RESOURCE_POLLING_INTERVAL,
# RESOURCE_START, and RESOURCE_UP_VALUE are used to specify Package
# Resource Dependencies. To define a package Resource Dependency, a
# RESOURCE_NAME line with a fully qualified resource path name, and
# one or more RESOURCE_UP_VALUE lines are required. The
# RESOURCE_POLLING_INTERVAL and the RESOURCE_START are optional.
#
# The RESOURCE_POLLING_INTERVAL indicates how often, in seconds, the
# resource is to be monitored. It will be defaulted to 60 seconds if
# RESOURCE_POLLING_INTERVAL is not specified.
#
# The RESOURCE_START option can be set to either AUTOMATIC or DEFERRED.
# The default setting for RESOURCE_START is AUTOMATIC. If AUTOMATIC
# is specified, Serviceguard will start up resource monitoring for
# these AUTOMATIC resources automatically when the node starts up.
# If DEFERRED is selected, Serviceguard will not attempt to start
# resource monitoring for these resources during node start up. User
# should specify all the DEFERRED resources in the package run script
# so that these DEFERRED resources will be started up from the package
# run script during package run time.
```

```

#
# RESOURCE_UP_VALUE requires an operator and a value. This defines
# the resource 'UP' condition. The operators are =, !=, >, <, >=,
# and <=, depending on the type of value. Values can be string or
# numeric. If the type is string, then only = and != are valid
# operators. If the string contains whitespace, it must be enclosed
# in quotes. String values are case sensitive. For example,
#
#                                     Resource is up when its value is
#                                     -----
#             RESOURCE_UP_VALUE      = UP                      "UP"
#             RESOURCE_UP_VALUE      != DOWN                  Any value except "DOWN"
#             RESOURCE_UP_VALUE      = "On Course"           "On Course"
#
# If the type is numeric, then it can specify a threshold, or a range to
# define a resource up condition. If it is a threshold, then any operator
# may be used. If a range is to be specified, then only > or >= may be used
# for the first operator, and only < or <= may be used for the second operator.
# For example,
#                                     Resource is up when its value is
#                                     -----
#             RESOURCE_UP_VALUE      = 5                      5          (threshold)
#             RESOURCE_UP_VALUE      > 5.1                   greater than 5.1 (threshold)
#             RESOURCE_UP_VALUE      > -5 and < 10        between -5 and 10 (range)
#
# Note that "and" is required between the lower limit and upper limit
# when specifying a range. The upper limit must be greater than the lower
# limit. If RESOURCE_UP_VALUE is repeated within a RESOURCE_NAME block, then
# they are inclusively OR'd together. Package Resource Dependencies may be
# defined by repeating the entire RESOURCE_NAME block.
#
# Example : RESOURCE_NAME          /net/interfaces/lan/status/lan0
#             RESOURCE_POLLING_INTERVAL 120
#             RESOURCE_START            AUTOMATIC
#             RESOURCE_UP_VALUE         = RUNNING
#             RESOURCE_UP_VALUE         = ONLINE
#
# Means that the value of resource /net/interfaces/lan/status/lan0
# will be checked every 120 seconds, and is considered to
# be 'up' when its value is "RUNNING" or "ONLINE".
#
# Uncomment the following lines to specify Package Resource Dependencies.
#
#RESOURCE_NAME          <Full_path_name>
#RESOURCE_POLLING_INTERVAL <numeric_seconds>
#RESOURCE_START          <AUTOMATIC/DEFERRED>
#RESOURCE_UP_VALUE        <op> <string_or_numeric> [and <op> <numeric>]
#
# Access Control Policy Parameters.
#
# Three entries set the access control policy for the package:
# First line must be USER_NAME, second USER_HOST, and third USER_ROLE.
# Enter a value after each.
#
# 1. USER_NAME can either be ANY_USER, or a maximum of
#    8 login names from the /etc/passwd file on user host.
# 2. USER_HOST is where the user can issue Serviceguard commands.
#    If using Serviceguard Manager, it is the COM server.
#    Choose one of these three values: ANY_SERVICEGUARD_NODE, or
#    (any) CLUSTER_MEMBER_NODE, or a specific node. For node,
#    use the official hostname from domain name server, and not
#    an IP addresses or fully qualified name.
# 3. USER_ROLE must be PACKAGE_ADMIN. This role grants permission

```

## Module 7

### Packages and Services

```
#      to MONITOR, plus for administrative commands for the package.  
#  
# These policies do not effect root users. Access Policies here  
# should not conflict with policies defined in the cluster configuration file.  
#  
# Example: to configure a role for user john from node noir to  
# administer the package, enter:  
# USER_NAME  john  
# USER_HOST  noir  
# USER_ROLE  PACKAGE_ADMIN
```

## 7-9. SLIDE: Step 2: Build Package Run/Halt Script

### Step 2: Build Package Control Script

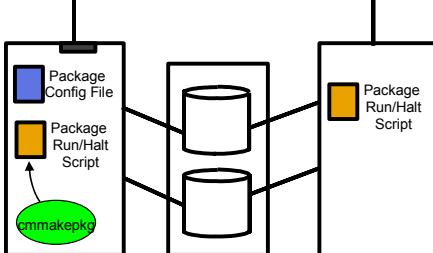


```
PATH=
VG[0]=
LV[0]="" ; FS[0]="" ; FS_MOUNT_OPT[0]=""

IP[0]=""
SUBNET[0]=""

SERVICE_NAME[0]=
SERVICE_CMD[0]="" 
SERVICE_RESTART[0]=""

function customer_defined_run_cmds
{
}
function customer_defined_halt_cmds
{}
```



**Step-by-Step Procedure to Package an Application**

1. Build the Package Configuration File
2. **Build the Package Run/Halt Script**
3. Compile and Distribute the Binary File

### Student Notes

Once the package configuration file is created, the package control script ( also known as the ( run / halt ) script ) needs to be created.

#### Step 2: Build the Package Run/Halt Script

The procedure for building the package control script is to create a template with the command:

```
# cmmakepkg -v -s package_script_name
```

The next step is to edit/customize the script with the `vi` editor:

```
# vi package_script_name
```

**The following are some of the major fields in the package control script:**

<i>PATH</i>	The value of the <code>PATH</code> variable used when executing the control script.
<i>VGCHANGE</i>	The name of the volume group activated exclusively by the control script.
<i>CVM_ACTIVATION_CMD</i>	This is reserved for future use with CVM.
<i>VG</i>	The name of the volume group activated exclusively by the control script.
<i>CVM_DG</i>	Cluster Volume Manager Disk Groups.
<i>VXVM_DG</i>	The name of the VxVM Disk Group used by the package.
<i>LV</i>	The name of the logical volume mounted by the control script.
<i>FS</i>	The name of the file system mount point used when mounting the logical volume.
<i>FS_MOUNT_OPT</i>	The list of mount options to be used when mounting the logical volume on the file system directory ( mount point ).
<i>FS_UNMOUNT_COUNT</i>	The number of unmount attempts for each file system.
<i>FS_MOUNT_RETRY_COUNT</i>	The number of mount retries for each file system.
<i>IP</i>	The IP address of the package. The IP and SUBNET lines are used in pairs. If the package will not be assigned an IP, there is no need to specify the SUBNET
<i>SUBNET</i>	The subnet of the package. This determines which subnet and interface the package will use to transfer data. (See the description for IP for further information on this parameter)
<i>SERVICE_NAME</i>	The name of the service process that is started. This is the link between the package configuration and control scripts. It must be exactly the same in both files. The only exception is that quotes are okay in the control script but <b>not</b> in the configuration script.
<i>SERVICE_CMD</i>	The full pathname of the command used to start the service process.

## *SERVICE\_RESTART*

The number of times to restart the service process if it fails.  
Valid configurations are:

- no restarts

SERVICE\_RESTART[ 0 ]= or

```
SERVICE_RESTART[0] = "-r 0"
```

- restart n-times on the same node

SERVICE\_RESTART[ 0 ]=“-r n” (n =number >= 0)

- always restart on the same node

SERVICE RESTART[ 0 ] = "-R"

### *customer\_defined\_run\_cmds*

Application-specific commands executed when starting the package. These commands run **before** services are started.

#### *customer\_defined\_halt\_cmds*

Application-specific commands executed when halting the package. These commands are run **after** services are halted.

## 7-10. TEXT PAGE: Sample Control Script File

### Special Note

The following package control script is from version 11.17 of Serviceguard.  
To conserve space in this course book, most of the functions in the second  
half of the script have had the code within the function removed.

```
# @(#) A.11.17.00 Date: 09/23/05 $  
# ****  
# *  
# *      HIGH AVAILABILITY PACKAGE CONTROL SCRIPT (template)  
# *  
# *      Note: This file MUST be edited before it can be used.  
# *  
# ****  
  
# The environment variables PACKAGE, NODE, SG_PACKAGE,  
# SG_NODE and SG_SCRIPT_LOG_FILE are set by Serviceguard  
# at the time the control script is executed.  
# Do not set these environment variables yourself!  
# The package may fail to start or halt if the values for  
# these environment variables are altered.  
  
# NOTE: Starting from 11.17, all environment variables set by  
# Serviceguard implicitly at the time the control script is  
# executed will contain the prefix "SG_". Do not set any variable  
# with the defined prefix, or the control script may not  
# function as it should.  
  
. ${SGCONFFILE:=/etc/cmcluster.conf}  
  
# UNCOMMENT the variables as you set them.  
  
# Set PATH to reference the appropriate directories.  
PATH=$SGSBIN:/usr/bin:/usr/sbin:/etc:/bin  
  
# VOLUME GROUP ACTIVATION:  
# Specify the method of activation for volume groups.  
# Leave the default ("VGCHANGE="vgchange -a e") if you want volume  
# groups activated in exclusive mode. This assumes the volume groups have  
# been initialized with 'vgchange -c y' at the time of creation.  
#  
# Uncomment the first line (VGCHANGE="vgchange -a e -q n"), and comment  
# out the default, if your disks are mirrored on separate physical paths,  
#  
# Uncomment the second line (VGCHANGE="vgchange -a e -q n -s"), and comment  
# out the default, if your disks are mirrored on separate physical paths,  
# and you want the mirror resynchronization to occur in parallel with  
# the package startup.  
#  
# Uncomment the third line (VGCHANGE="vgchange -a y") if you wish to  
# use non-exclusive activation mode. Single node cluster configurations
```

```
# must use non-exclusive activation.  
#  
# VGCHANGE="vgchange -a e -q n"  
# VGCHANGE="vgchange -a e -q n -s"  
# VGCHANGE="vgchange -a y"  
VGCHANGE="vgchange -a e"                      # Default  
  
# CVM DISK GROUP ACTIVATION:  
# Specify the method of activation for CVM disk groups.  
# Leave the default  
# (CVM_ACTIVATION_CMD="vxldg -g \$DiskGroup set activation=exclusivewrite")  
# if you want disk groups activated in the exclusive write mode.  
#  
# Uncomment the first line  
# (CVM_ACTIVATION_CMD="vxldg -g \$DiskGroup set activation=readonly"),  
# and comment out the default, if you want disk groups activated in  
# the readonly mode.  
#  
# Uncomment the second line  
# (CVM_ACTIVATION_CMD="vxldg -g \$DiskGroup set activation=sharedread"),  
# and comment out the default, if you want disk groups activated in the  
# shared read mode.  
#  
# Uncomment the third line  
# (CVM_ACTIVATION_CMD="vxldg -g \$DiskGroup set activation=sharedwrite"),  
# and comment out the default, if you want disk groups activated in the  
# shared write mode.  
#  
# CVM_ACTIVATION_CMD="vxldg -g \$DiskGroup set activation=readonly"  
# CVM_ACTIVATION_CMD="vxldg -g \$DiskGroup set activation=sharedread"  
# CVM_ACTIVATION_CMD="vxldg -g \$DiskGroup set activation=sharedwrite"  
CVM_ACTIVATION_CMD="vxldg -g \$DiskGroup set activation=exclusivewrite"  
  
# VOLUME GROUPS  
# Specify which volume groups are used by this package. Uncomment VG[0]=""  
# and fill in the name of your first volume group. You must begin with  
# VG[0], and increment the list in sequence.  
#  
# For example, if this package uses your volume groups vg01 and vg02, enter:  
#      VG[0]=vg01  
#      VG[1]=vg02  
#  
# The volume group activation method is defined above. The filesystems  
# associated with these volume groups are specified below.  
#  
#VG[0]=""  
  
# CVM DISK GROUPS  
# Specify which cvm disk groups are used by this package. Uncomment  
# CVM_DG[0]="" and fill in the name of your first disk group. You must  
# begin with CVM_DG[0], and increment the list in sequence.  
#  
# For example, if this package uses your disk groups dg01 and dg02, enter:  
#      CVM_DG[0]=dg01  
#      CVM_DG[1]=dg02  
#  
# The cvm disk group activation method is defined above. The filesystems
```

## Module 7

### Packages and Services

```
# associated with these volume groups are specified below in the CVM_*
# variables.
#
#CVM_DG[0]="""

# NOTE: Do not use CVM and VxVM disk group parameters to reference
# devices used by CFS (cluster file system). CFS resources are
# controlled by the Disk Group and Mount Multi-node packages.
#
# VxVM DISK GROUPS
# Specify which VxVM disk groups are used by this package. Uncomment
# VXVM_DG[0]="" and fill in the name of your first disk group. You must
# begin with VXVM_DG[0], and increment the list in sequence.
#
# For example, if this package uses your disk groups dg01 and dg02, enter:
#     VXVM_DG[0]=dg01
#     VXVM_DG[1]=dg02
#
# The cvm disk group activation method is defined above.
#
#VXVM_DG[0]=""

#
# NOTE: A package could have LVM volume groups, CVM disk groups and VxVM
# disk groups.
#
# NOTE: When VxVM is initialized it will store the hostname of the
# local node in its volboot file in a variable called 'hostid'.
# The Serviceguard package control scripts use both the values of
# the hostname(1m) command and the VxVM hostid. As a result
# the VxVM hostid should always match the value of the
# hostname(1m) command.
#
# If you modify the local host name after VxVM has been
# initialized and such that hostname(1m) does not equal uname -n,
# you need to use the vxdctl(1m) command to set the VxVM hostid
# field to the value of hostname(1m). Failure to do so will
# result in the package failing to start.

# VOLUME GROUP AND DISK GROUP DEACTIVATION RETRY COUNT
# Specify the number of deactivation retries for each disk group and volume
# group at package shutdown. The default is 0.
DEACTIVATION_RETRY_COUNT=0

#
# RAW DEVICES
# If you are using raw devices for your application, this parameter allows
# you to specify if you want to kill the processes that are accessing the
# raw devices at package halt time. If raw devices are still being accessed
# at package halt time, volume group or disk group deactivation can fail,
# causing the package halt to also fail. This problem usually happens when
# the application does not shut down properly.
# Note that if you are using Oracle's Cluster Ready Service, killing this
# service could cause the node to reboot.
# The legal values are "YES" and "NO". The default value is "NO".
# The value that is set for this parameter affects all raw devices associated
# with the LVM volume groups and CVM disk groups defined in the package.
```

```
KILL_PROCESSES_ACCESSING_RAW_DEVICES="NO"

# FILESYSTEMS
# Filesystems are defined as entries specifying the logical volume, the
# mount point, the mount, umount and fsck options and type of the file system.
# Each filesystem will be fsck'd prior to being mounted. The filesystems
# will be mounted in the order specified during package startup and will
# be unmounted in reverse order during package shutdown. Ensure that
# volume groups referenced by the logical volume definitions below are
# included in volume group definitions above.
#
# Specify the filesystems which are used by this package. Uncomment
# LV[0]="" ; FS[0]="" ; FS_MOUNT_OPT[0]="" ; FS_UNMOUNT_OPT[0]="" ;
FS_FSCK_OPT[0]=""
# FS_TYPE[0]="" and fill in the name of your first logical volume,
# filesystem, mount, umount and fsck options and filesystem type
# for the file system. You must begin with LV[0], FS[0],
# FS_MOUNT_OPT[0], FS_UNMOUNT_OPT[0], FS_FSCK_OPT[0], FS_TYPE[0]
# and increment the list in sequence.
#
# Note: The FS_TYPE parameter lets you specify the type of filesystem to be
# mounted. Specifying a particular FS_TYPE will improve package failover time.
# The FSCK_OPT and FS_UNMOUNT_OPT parameters can be used to include the
# -s option with the fsck and umount commands to improve performance for
# environments that use a large number of filesystems. (An example of a
# large environment is given below following the description of the
# CONCURRENT_MOUNT_AND_UNMOUNT_OPERATIONS parameter.)
#
# Example: If a package uses two JFS filesystems, pkg01a and pkg01b,
# which are mounted on LVM logical volumes lvol1 and lvol2 for read and
# write operation, you would enter the following:
#     LV[0]=/dev/vg01/lvol1; FS[0]=/pkg01a; FS_MOUNT_OPT[0]="-o rw";
#     FS_UNMOUNT_OPT[0]="" ; FS_FSCK_OPT[0]="" ; FS_TYPE[0]="vxfs"
#
#     LV[1]=/dev/vg01/lvol2; FS[1]=/pkg01b; FS_MOUNT_OPT[1]="-o rw"
#     FS_UNMOUNT_OPT[1]="" ; FS_FSCK_OPT[1]="" ; FS_TYPE[1]="vxfs"
#
#LV[0]="" ; FS[0]="" ; FS_MOUNT_OPT[0]="" ; FS_UNMOUNT_OPT[0]="" ;
FS_FSCK_OPT[0]=""
#FS_TYPE[0]=""
#
# VOLUME RECOVERY
#
# When mirrored VxVM volumes are started during the package control
# bring up, if recovery is required the default behavior is for
# the package control script to wait until recovery has been
# completed.
#
# To allow mirror resynchronization to occur in parallel with
# the package startup, uncomment the line
# VXVOL="vxvol -g \$DiskGroup -o bg startall" and comment out the default.
#
# VXVOL="vxvol -g \$DiskGroup -o bg startall"
VXVOL="vxvol -g \$DiskGroup startall"      # Default

# FILESYSTEM UNMOUNT COUNT
# Specify the number of umount attempts for each filesystem during package
```

Module 7  
**Packages and Services**

```
# shutdown. The default is set to 1.  
FS_UMOUNT_COUNT=1  
  
# FILESYSTEM MOUNT RETRY COUNT.  
# Specify the number of mount retries for each filesystem.  
# The default is 0. During startup, if a mount point is busy  
# and FS_MOUNT_RETRY_COUNT is 0, package startup will fail and  
# the script will exit with 1. If a mount point is busy and  
# FS_MOUNT_RETRY_COUNT is greater than 0, the script will attempt  
# to kill the user responsible for the busy mount point  
# and then mount the file system. It will attempt to kill user and  
# retry mount, for the number of times specified in FS_MOUNT_RETRY_COUNT.  
# If the mount still fails after this number of attempts, the script  
# will exit with 1.  
# NOTE: If the FS_MOUNT_RETRY_COUNT > 0, the script will execute  
# "fuser -ku" to freeup busy mount point.  
FS_MOUNT_RETRY_COUNT=0  
  
#  
# Configuring the concurrent operations below can be used to improve the  
# performance for starting up or halting a package. The maximum value for  
# each concurrent operation parameter is 1024. Set these values carefully.  
# The performance could actually decrease if the values are set too high  
# for the system resources available on your cluster nodes. Some examples  
# of system resources that can affect the optimum number of concurrent  
# operations are: number of CPUs, amount of available memory, the kernel  
# configuration for nfile and nproc. In some cases, if you set the number  
# of concurrent operations too high, the package may not be able to start  
# or to halt. For example, if you set CONCURRENT_VGCHANGE_OPERATIONS=5  
# and the node where the package is started has only one processor, then  
# running concurrent volume group activations will not be beneficial.  
# It is suggested that the number of concurrent operations be tuned  
# carefully, increasing the values a little at a time and observing the  
# effect on the performance, and the values should never be set to a value  
# where the performance levels off or declines. Additionally, the values  
# used should take into account the node with the least resources in the  
# cluster, and how many other packages may be running on the node.  
# For instance, if you tune the concurrent operations for a package so  
# that it provides optimum performance for the package on a node while  
# no other packages are running on that node, the package performance  
# may be significantly reduced, or may even fail when other packages are  
# already running on that node.  
#  
# CONCURRENT VGCHANGE OPERATIONS  
# Specify the number of concurrent volume group activations or  
# deactivations to allow during package startup or shutdown.  
# Setting this value to an appropriate number may improve the performance  
# while activating or deactivating a large number of volume groups in the  
# package. If the specified value is less than 1, the script defaults it  
# to 1 and proceeds with a warning message in the package control script  
# logfile.  
CONCURRENT_VGCHANGE_OPERATIONS=1  
  
# CONCURRENT FSCK OPERATIONS  
# Specify the number of concurrent fsck to allow during package startup.  
# Setting this value to an appropriate number may improve the performance  
# while checking a large number of file systems in the package. If the
```

```
# specified value is less than 1, the script defaults it to 1 and proceeds
# with a warning message in the package control script logfile.
CONCURRENT_FSCK_OPERATIONS=1

# CONCURRENT MOUNT AND UMOUNT OPERATIONS
# Specify the number of concurrent mounts and umounts to allow during
# package startup or shutdown.
# Setting this value to an appropriate number may improve the performance
# while mounting or un-mounting a large number of file systems in the package.
# If the specified value is less than 1, the script defaults it to 1 and
# proceeds with a warning message in the package control script logfile.
CONCURRENT_MOUNT_AND_UMOUNT_OPERATIONS=1

# Example: If a package uses 50 JFS filesystems, pkg01aa through pkg01bx,
# which are mounted on the 50 logical volumes lvvol1..lvvol50 for read and write
# operation, you may enter the following:
#
#      CONCURRENT_FSCK_OPERATIONS=50
#      CONCURRENT_MOUNT_AND_UMOUNT_OPERATIONS=50
#
#      LV[0]=/dev/vg01/lvvol1; FS[0]=/pkg01aa; FS_MOUNT_OPT[0]="-o rw";
#      FS_UMOUNT_OPT[0]="-s"; FS_FSCK_OPT[0]="-s"; FS_TYPE[0]="vxfs"
#
#      LV[1]=/dev/vg01/lvvol2; FS[1]=/pkg01ab; FS_MOUNT_OPT[1]="-o rw"
#      FS_UMOUNT_OPT[1]="-s"; FS_FSCK_OPT[1]="-s"; FS_TYPE[0]="vxfs"
#      :
#      :
#      :
#      LV[49]=/dev/vg01/lvvol50; FS[49]=/pkg01bx; FS_MOUNT_OPT[49]="-o rw"
#      FS_UMOUNT_OPT[49]="-s"; FS_FSCK_OPT[49]="-s"; FS_TYPE[0]="vxfs"
#
# IP ADDRESSES
# Specify the IP and Subnet address pairs which are used by this package.
# You could specify IPv4 or IPv6 IP and subnet address pairs.
# Uncomment IP[0]="" and SUBNET[0]="" and fill in the name of your first
# IP and subnet address. You must begin with IP[0] and SUBNET[0] and
# increment the list in sequence.
#
# For example, if this package uses an IP of 192.10.25.12 and a subnet of
# 192.10.25.0 enter:
#      IP[0]=192.10.25.12
#      SUBNET[0]=192.10.25.0
#      (netmask=255.255.255.0)
#
# Hint: Run "netstat -i" to see the available subnets in the Network field.
#
# For example, if this package uses an IPv6 IP of 2001::1/64
# The address prefix identifies the subnet as 2001::/64 which is an available
# subnet.
# enter:
#      IP[0]=2001::1
#      SUBNET[0]=2001::/64
#      (netmask=ffff:ffff:ffff:ffff::)
# Alternatively the IPv6 IP/Subnet pair can be specified without the prefix
# for the IPv6 subnet.
#      IP[0]=2001::1
#      SUBNET[0]=2001::
```

Module 7  
**Packages and Services**

```
#           (netmask=ffff:ffff:ffff:ffff::)
#
# Hint: Run "netstat -i" to see the available IPv6 subnets by looking
# at the address prefixes
# IP/Subnet address pairs for each IP address you want to add to a subnet
# interface card. Must be set in pairs, even for IP addresses on the same
# subnet.
#
#IP[0]=""
#SUBNET[0]=""

# SERVICE NAMES AND COMMANDS.
# Specify the service name, command, and restart parameters which are
# used by this package. Uncomment SERVICE_NAME[0]="", SERVICE_CMD[0]="" ,
# SERVICE_RESTART[0]="" and fill in the name of the first service, command,
# and restart parameters. You must begin with SERVICE_NAME[0], SERVICE_CMD[0] ,
# and SERVICE_RESTART[0] and increment the list in sequence.
#
# For example:
#   SERVICE_NAME[0]=pkg1a
#   SERVICE_CMD[0]="/usr/bin/X11/xclock -display 192.10.25.54:0"
#   SERVICE_RESTART[0]="" # Will not restart the service.
#
#   SERVICE_NAME[1]=pkg1b
#   SERVICE_CMD[1]="/usr/bin/X11/xload -display 192.10.25.54:0"
#   SERVICE_RESTART[1]="-r 2" # Will restart the service twice.
#
#   SERVICE_NAME[2]=pkg1c
#   SERVICE_CMD[2]="/usr/sbin/ping"
#   SERVICE_RESTART[2]="-R" # Will restart the service an infinite
#                           number of times.
#
# Note: No environmental variables will be passed to the command, this
# includes the PATH variable. Absolute path names are required for the
# service command definition. Default shell is /usr/bin/sh.
#
#SERVICE_NAME[0]=""
#SERVICE_CMD[0]=""
#SERVICE_RESTART[0]=""

# DEFERRED_RESOURCE NAME
# Specify the full path name of the 'DEFERRED' resources configured for
# this package. Uncomment DEFERRED_RESOURCE_NAME[0]="" and fill in the
# full path name of the resource.
#
#DEFERRED_RESOURCE_NAME[0]=""

# DTC manager information for each DTC.
# Example: DTC[0]=dtc_20
#DTC_NAME[0]=""

# HA_NFS_SCRIPT_EXTENSION
# If the package uses HA NFS, this variable can be used to alter the
# name of the HA NFS script. If not set, the name of this script is
# assumed to be "ha_nfs.sh". If set, the "sh" portion of the default
# script name is replaced by the value of this variable. So if
# HA_NFS_SCRIPT_EXTENSION is set to "package1.sh", for example, the name
```

```
# of the HA NFS script becomes "ha_nfs.package1.sh". In any case,  
# the HA NFS script must be placed in the same directory as the package  
# control script. This allows multiple packages to be run out of the  
# same directory, as needed by SGeSAP.  
#HA_NFS_SCRIPT_EXTENSION=""  
  
# START OF CUSTOMER DEFINED FUNCTIONS  
  
# This function is a place holder for customer define functions.  
# You should define all actions you want to happen here, before the service is  
# started. You can create as many functions as you need.  
  
function customer_defined_run_cmds  
{  
# ADD customer defined run commands.  
: # do nothing instruction, because a function must contain some command.  
  
    test_return 51  
}  
  
# This function is a place holder for customer define functions.  
# You should define all actions you want to happen here, after the service is  
# halted.  
  
function customer_defined_halt_cmds  
{  
# ADD customer defined halt commands.  
: # do nothing instruction, because a function must contain some command.  
    test_return 52  
}  
  
# END OF CUSTOMER DEFINED FUNCTIONS  
  
# START OF RUN FUNCTIONS  
  
function verify_physical_data_replication  
{  
}  
  
function verify_ha_nfs  
{  
}  
  
function ha_nfs_file_locks  
{  
}  
  
function activate_volume_group  
{  
}  
  
function activate_disk_group
```

**Module 7**  
**Packages and Services**

```
{  
}  
  
function check_dg  
{  
}  
  
function freeup_busy_mountpoint_and_mount_fs  
{  
}  
  
function check_vxvm_vol_available  
{  
}  
  
function check_and_mount  
{  
}  
  
function retry_print  
{  
}  
  
function add_ip_address  
{  
}  
  
function get_ownership_dtc  
{  
}  
  
function start_services  
{  
}  
  
# END OF RUN FUNCTIONS.  
  
# START OF HALT FUNCTIONS  
  
function stop_resources  
{  
}  
  
function halt_services  
{
```

```
}
```

```
function disown_dtc
{}
```

```
function remove_ip_address
{}
```

```
function umount_fs
{}
```

```
function deactivate_volume_group
{}
```

```
function deactivate_disk_group
{}
```

```
# END OF HALT FUNCTIONS.
```

```
# FUNCTIONS COMMON TO BOTH RUN AND HALT.
```

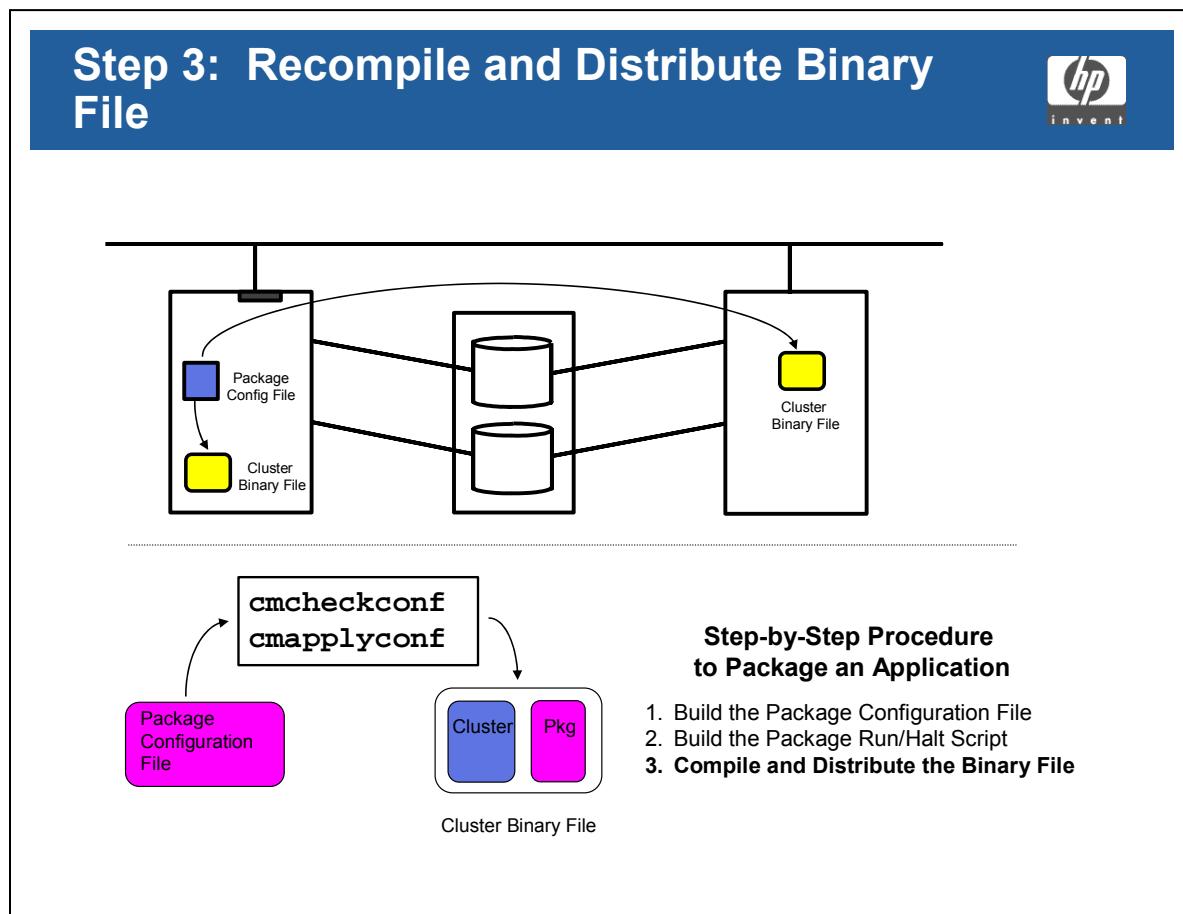
```
function test_return
{}
```

```
# END OF FUNCTIONS COMMON TO BOTH RUN AND HALT
```

```
-----MAINLINE Control Script Code Starts Here-----
```

```
-----MAINLINE Control Script Code Ends Here-----
```

## 7-11. SLIDE: Step 3: Recompile and Distribute Binary File



### Student Notes

The final step in the package creation process is to rebuild the cluster binary file to include the package information.

#### Step 3: Recompile and Distribute the Binary File

The commands to verify the package files and integrate the package information into the cluster binary file are:

```
# cmcheckconf -v -P package_configuration_file  
# cmapplyconf -v -P package_configuration_file
```

Note that the package information is *added* to the cluster binary file. The cluster binary file no longer needs to be rebuilt from scratch ( starting with MC/ServiceGuard release 10.10 ).

## 7-12. SLIDE: Package Scripts

### Package Scripts



#### Package Run Script

- Script called by Serviceguard to start the application
- Core functionality provided by HP in the form of a template
- Customer edits template to be specific for their application
- One run script per package; script must be located on each node in the cluster

#### Package Halt Script

- Script called by Serviceguard to halt the application
- This script is normally the same script as the run script

```
package_script start  ( starts the application – never executed manually )
package_script stop   ( stops the application – never executed manually )
```

### Student Notes

The package run / halt script contains all the information to bring up and bring down the application completely. It is highly recommended that these two scripts be the same file (referred to as a package “control script”), and the behavior of the script be determined by the first passed parameter.

The package control script template is created with the `cmmakepkg` command:

```
cmmakepkg -v -s pkg.cntl
```

Once the control script template is created, it must be customized for the specific package.

One of the parameters specified in the package control script is `SERVICE_CMD`. This command (which will be a service process) will be spawned and monitored by Serviceguard in the following way:

- The command will be spawned by `cmrunserv`. Once spawned, the resulting PID will be monitored every 10 seconds.

Module 7  
**Packages and Services**

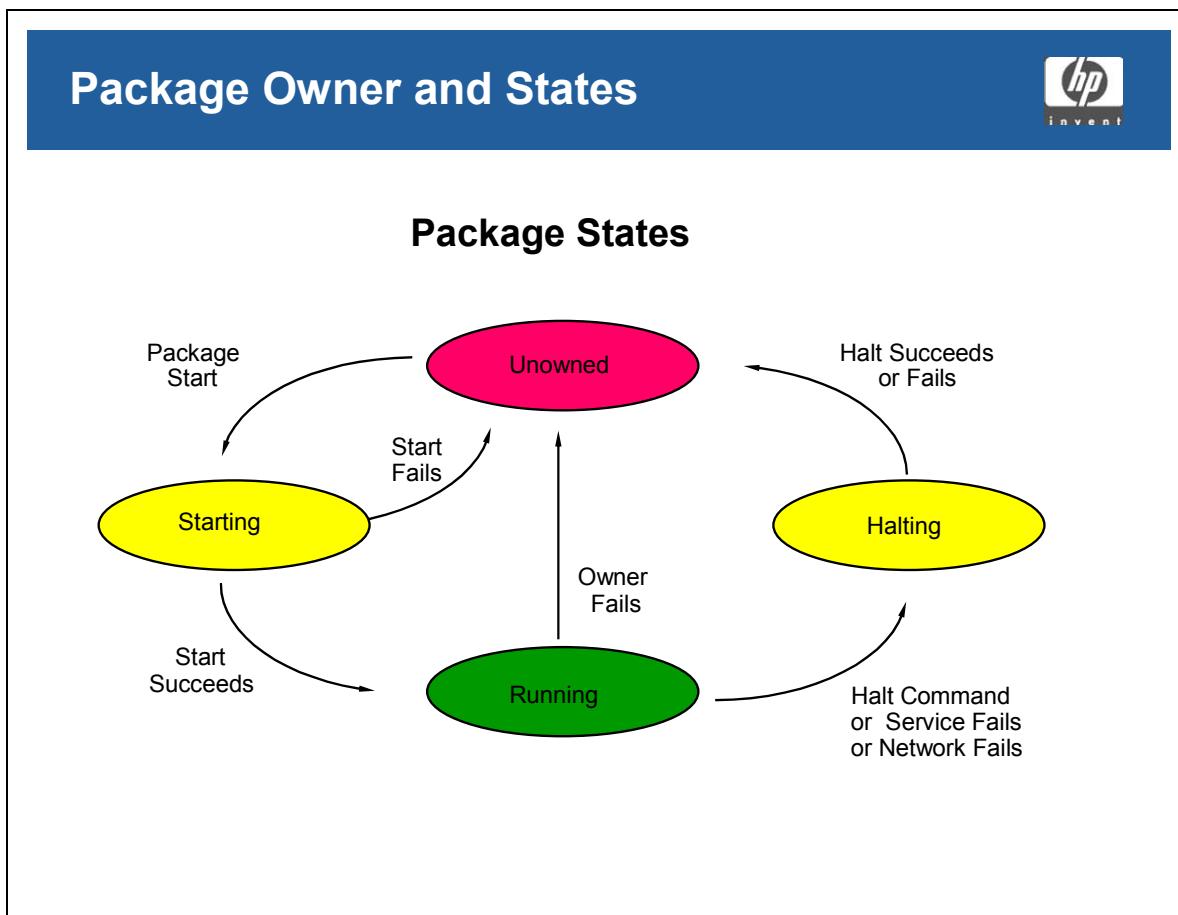
- If the PID exits or terminates, Serviceguard responds with a service failure and takes appropriate actions.
- If SERVICE\_CMD is a shell script that starts other processes (like database processes) and then exits, Serviceguard will see this normal exit as a failure and Serviceguard will respond accordingly.

---

**NOTE:** Serviceguard's mission is to keep persistent processes persisting.  
Therefore, SERVICE\_CMD must be a persistent process.

---

## 7-13. SLIDE: Package Owner and State



### Student Notes

#### Package Owner

The owner of a package is the node that is currently executing the package.

- The coordinator determines which node will be the owner for all the packages. The current package switching state and node switching states are used to determine the owner. If no owner can be assigned, then the owner is set to `NO_OWNER`.
- If a package owner fails, the coordinator sets the owner to `NO_OWNER` and immediately begins to look for a new owner.

#### Package State

The package state is used to indicate the health and status of a package. There are four possible package states:

**Unowned** This state indicates the package is NOT running on any node and no attempt is being made to start up the package. A package would be in this state if package

Module 7  
**Packages and Services**

switching is disabled, or node switching has been disabled for all nodes related to this package.

Starting	This state indicates the package run script is currently being executed in an attempt to bring up the package. Packages do not stay in this state for very long since this is a transition state between Unowned and Running.
Running	This state indicates the package is up and running. This will be the state for most packages within a healthy cluster.
Halting	This state indicates the halt script is being executed to bring down the package on the current machine. Once the halt script completes, the package will transition into an Unowned state and the coordinator will try to find a new owner ( if one exists depending on the package and node switching states ). If the package has been halted manually with the cmhalt command, the package will not be started on another node, even if package switching is enabled and a node is enabled to accept it.

## **Package Manager**

Package management is performed by the package manager instance on each node in the following types of clusters: (1) a Serviceguard cluster, (2) a ServiceGuard OPS edition cluster, or (3) a Serviceguard with the Serviceguard Extension for RAC cluster. Each package has an *owner*, a *state*, and *attributes* associated with it. It is the owning node's responsibility to keep this package information and report it as needed to the coordinator. The package manager instance, which owns a particular package, is referred to as the *package owner*.

## **Package Coordinator**

Package management is coordinated among nodes by the package manager instance, which resides on the cluster coordinator node. This package manager instance is referred to as the *package coordinator*.

The package coordinator –

- is on the same node as the *Cluster Coordinator*.
- directs package manipulation for all packages. The package manager instance, which owns a package, performs the actual work.
- keeps the information on the owner, state, and attributes and uses it to perform package management.
- must pass the information on to all nodes in the cluster. This allows global package information to persist across reboots.

---

**NOTE:** All nodes must have the same information to compensate for failures of the owning node or cluster coordinator.

---

## 7-14. SLIDE: Package Configuration Procedure

### Package Configuration Procedure



1. `mkdir /etc/cmcluster/package_name` ( Create directory for package specific files )
2. `cd /etc/cmcluster/package_name` ( Change to the directory )
3. `cmmakepkg -v -p pkg.conf` ( Create package configuration template )
4. `vi pkg.conf` ( Modify package parameters )
5. `cmmakepkg -v -s pkg.cntl` ( Create package control script template )
6. `vi pkg.cntl` ( Modify script variables )
7. `ftp pkg.cntl to other systems in the cluster` ( Distribute the control script to all nodes )
8. `cmcheckconf -v -P /etc/cmcluster/package_name/pkg.conf`  
( Verify the package configuration file )
9. `cmapplyconf -v -P /etc/cmcluster/package_name/pkg.conf`  
( Compile package into cluster binary file )

### Student Notes

The slide shows the basic steps for configuring a package to run in a Serviceguard environment.

#### (Steps 1-4) Create the Package Configuration File

The files related to a particular package should be kept together in a subdirectory under `/etc/cmcluster`. The first step in configuring a package is to create a subdirectory for the package, and then create the package configuration file in this subdirectory.

Use `cmmakepkg -v -p pkg_config_file` to create the package configuration file.

#### (Steps 5 and 6) Create the Package Control Script

Once the package configuration file is created, the package control script needs to be created using the same approach. First use `cmmakepkg -v -s pkg_control_file` to create a template script, then modify the template to be specific for the application.

#### (Step 7) Copy Package Control Script and Other Files to Other Nodes

Serviceguard does not distribute the packaging related ASCII files to the other nodes in the cluster. This task must be done by the system administrator. The system administrator

Module 7  
**Packages and Services**

should copy the package control script and any other package-related files ( like an application monitor script, if being used ) to the other nodes within the cluster. The packaging files should be kept in subdirectories by exactly the same pathnames using the same names.

**(Steps 8 and 9) Create and Distribute the Binary File**

Once the package files have been customized and distributed, the files should be checked (in conjunction with the cluster configuration file) for syntax and logic errors. The `cmcheckconf` command performs these checks. `cmcheckconf` is monolithic in that it doesn't check syntax among packages. This means `cmcheckconf` won't flag multiple packages that are configured with the same resources (volume groups are a particular problem, for example).

Specifically, `cmcheckconf` checks the following with respect to packages:

- Valid package name (and that at least one NODE\_NAME entry is included).
- There are no duplicate parameter entries.
- Values for parameters are within permitted ranges.
- Run and halt scripts exist on all nodes in the cluster and are executable
- Configured resources are available on all cluster nodes

Once the files have been verified to contain no errors, the `cmapplyconf` command is used to create and distribute a new binary file.

## 7-15. SLIDE: The Package Script Log File

### The Package Script Log File



- The default name of the Package Script log file is:  
`/etc/cmcluster/package_name/pkg.cntl.log`
- The name is typically the control script name with a ".log" suffix.
- There is one log file per package.
- The log file contains high level descriptions of major actions performed by the control script.
- More helpful than `syslog` file for package troubleshooting

### Student Notes

For each package control script, a corresponding package control script log file will be automatically created the first time the script is executed.

This log file contains information related to the starting and halting of the package. The log file provides detailed information on activities performed by the script (such as VG activations and IP address assignments) for each invocation of the package control script.

By default, the name of the package control script log file is  
`/etc/cmcluster/pkg_name/pkg_name.cntl.log`.

Beginning with A.11.17, it is possible to configure a package log file with a different location and/or a different name.

See the `#SCRIPT_LOG_FILE` line in the `package.conf` file for instructions to change the default package log file name and location.

## 7-16. SLIDE: Review of Commands for Controlling a Cluster

### Review of Commands for Controlling a Cluster



#### Package Management Commands

cmrunpkg  
cmhaltpkg  
cmmodpkg

#### Cluster Management Commands

cmrunc1  
cmhaltcl  
cmviewcl

#### Node Management Commands

cmrunnode  
cmhaltnode

### Student Notes

Package management, cluster management, and node management commands are used to manage the cluster once the binary file has been created and distributed.

#### Package Management Commands

cmrunpkg [-v] [-n NodeName] PackageName  
[ -v ] Verbose output  
[ -n ] Run package on a specific node

cmhaltpkg [-v] [-n NodeName] PackageName  
[ -v ] Verbose output  
[ -n ] Halt package on a specific node  
(if package is not running on "NodeName", it will not be halted)

cmmodpkg [-v] {-e|-d} [-n NodeName] . . . PackageName  
[ -v ] Verbose output  
-e Enable  
-d Disable  
[ -n ] Modify package for a specific node - else global package attributes are modified

## Cluster Management Commands

```
cmrunc1 [-v] [-f] [-n NodeName]
[ -v ] Verbose output
[ -f ] Force cluster to start up without warning messages
[ -n ] Run cmcld on a specific node

cmhaltcl [-v] [-f]
[ -v ] Verbose output
[ -f ] Force cluster to shutdown even if packages are running. This option will cause
      the halt script to be executed for running packages.

cmviewcl [-v] [-n NodeName] [-p PackageName] [-l {pkg|cluster|node}]
[ -v ] Verbose output
[ -n ] View information only about the specific node
[ -p ] View information only about the specific package
[ -l ] pkg | cluster | node Display only package, cluster, or node
                                specific information
```

---

**WARNING:** Serviceguard cannot guarantee data integrity if you try to start a cluster with the `cmrunc1 -f -n` command. These options should be used only in troubleshooting situations; they should *not* be used in a production environment.

---

## Node Management Commands

```
cmrunnode [-v] [NodeName]
[ -v ] Verbose output

cmhaltnode [-v] [-f] [NodeName]
[ -v ] Verbose output
[ -f ] Force node to shutdown even if packages are running on the node. This option
      will cause the halt script to execute for running packages.
```

## **7-17. REVIEW: Check Your Understanding**

## Directions

## Answer the following questions.

1. Where should the application executables reside? A single copy on the shared disks along with the application database and other dynamic data files; or two copies, one on each server in a non-shared (private) volume group?
  2. Should the shared file systems have entries in `/etc/fstab`? Why not?
  3. Should the shared volume groups be activated when the system boots (which happens by default)? How can this be prevented?
  4. How can you view the contents of the shared data with the cluster down? With the package down? In other words, how do you gain access to the database to do database maintenance, etc, and insure no users are accessing the application (either by taking down the package or the cluster)?
  5. If we kill an application on `node1` it automatically restarts on `node2`, but when we kill it on `node2` it does not automatically restart on `node1`. Why not? What must we do so it can switch back?

6. How can we manually move a single package from node1 to node2?
  
  
  
  
  
  
  
  
  
  
  
  
7. How can we take one node out of service ( and move applications/packages to adoptive nodes)? Return it to service? Will packages automatically return to primary node?
  
  
  
  
  
  
  
  
  
  
  
  
8. How can we stop one application, for example, for maintenance? How do we restart it?
  
  
  
  
  
  
  
  
  
  
  
  
9. How does Serviceguard detect node failures, network failures, and application failures?
  
  
  
  
  
  
  
  
  
  
  
  
10. Can we perform database maintenance with the cluster down?
  
  
  
  
  
  
  
  
  
  
  
  
11. Why does Serviceguard require all nodes to be present when the cluster is first formed? How can we override this? Would we want to?

## Module 7

# Packages and Services

12. Under what circumstances is a floating IP address beneficial/necessary? Would this IP address be added to your DNS server? What host name should be used? Can an application/package have multiple IP addresses? Would this be of value?
  13. How would you disable lower priority applications during a failover period? Notify administrators?
  14. If application, server, disk, router, SCSI card, network card, etc fails, it should be transparent to the user. What if it is also transparent to the administrator? How can we insure against this?

---

## 7-18. LAB: Package Configuration #1: logprog Package

### Directions

In this lab, we will configure a package called `logprog` to run in our cluster.

---

**NOTE:** We will add additional features to this package in later modules.

---

### Part 1: Compile our application.

#### On Node1

1. The source code (`logprog.c`) for your application can be found in the `/labs` directory. Compile and save it as `/usr/lbin/logprog`. This will be the application around which we will build our package.

When this program runs, it will echo a status message to `/tmp/logprog.out` once every 3 seconds.

Compile your program as follows:

```
# cd /labs; cc logprog.c -o /usr/lbin/logprog
```

2. In one window, execute your `logprog` program. Open another window and verify the output with `tail -f /tmp/logprog.out`

```
Window1# /usr/lbin/logprog
Window2# tail -f /tmp/logprog.out
```

3. After verifying that `logprog` operates as expected, press `<ctrl>c` to stop it.

## Part 2: Create and Edit the Package Configuration File

### On Node1

1. Make a directory for the logprog package files.
2. Change directory to the logprog package directory that was just created. Once there, create the package configuration file template.
3. Next, edit the package configuration file template.

```
# vi logprog.conf
```

and modify the following parameters:

PACKAGE_NAME	logprog
NODE_NAME	<node1>
NODE_NAME	<node2>
RUN_SCRIPT	/etc/cmcluster/logprog/logprog.cntl
RUN_SCRIPT_TIMEOUT	NO_TIMEOUT
HALT_SCRIPT	/etc/cmcluster/logprog/logprog.cntl
HALT_SCRIPT_TIMEOUT	NO_TIMEOUT
SERVICE_NAME	logprog_service
SUBNET	X.X.X.X     ← Use netstat -in to obtain the 10-net subnet

4. Save the changes and exit the file.

### **Part 3: Create and Edit the Package Control Script, and Distribute this Script to All Nodes**

### On Node1

1. First, we create the package control script template.

2. Secondly, edit the package control script template. Remember that this file is a script and that all variable assignments cannot contain any spaces.

```
# vi logprog.cntl
```

and modify the following parameters:

```
SERVICE_NAME[ 0 ]="logprog_service"
SERVICE_CMD[ 0 ]="/usr/lbin/logprog"
SERVICE_RESTART[ 0 ]="-r 2"
```

3. Save changes and exit the file.

4. Next, on `Node2`, create a subdirectory for your `logprog` package.

- Now, back on `Node1`, make the `logprog` control script executable, and copy it to `Node2` within the cluster.

6. Copy your new /usr/lbin/logprog program to the same directory on Node2.

## **Part 4: Create and Distribute the Binary File, and Start the Package**

## On Node 1

1. Check the package configuration file for errors and fix as needed.
  2. Since the configuration files are okay, apply the new configuration, thus compiling and distributing a new cluster binary file.
  3. View the package status.  
The package should be halted, but enabled to run on either node.
  4. Start the package.
  5. We can test the `logprog` package behavior by doing the following:

```
# cmhaltpkg -v logprog  
# cmviewcl -vp logprog  
  
# cmrumpkg -v -n <adoptive_node> logprog  
# cmmodpkg -v -e logprog  
  
# cmviewcl -vp logprog
```

6. We can further test our package by killing the logprog PID. We will kill the logprog process PID 3 times; the first two times because we specified 2 retries, and the third time to force a failover.

In the previous step, we moved the package to the `<adoptive_node>`. Execute the following steps on `<adoptive_node>` to force a failover back to `<primary_node>`.

```
# ps -ef | grep logprog           ← Kill the logprog process for the first time.  
# kill <pid>  
  
# ps -ef | grep logprog           ← Kill the logprog process for the second time.  
# kill <pid>  
  
# ps -ef | grep logprog           ← Kill the logprog process for the third time.  
# kill <pid>  
  
# cmviewcl -v                    ← The logprog package restarts on <primary node>.  
  
# cmmmodpkg -v -e -n <adoptive_node> logprog  
  
# cmviewcl -v                    ← logprog is now running on <primary node>.
```

- The package should now be running on <primary\_node>. Execute the following steps on <primary\_node> to force a failover back to <adoptive\_node>.

```
# ps -ef | grep logprog          ← Kill the logprog process for the first time.  
# kill <pid>  
# ps -ef | grep logprog          ← The process should restart on the same node.  
# kill <pid>  
  
# ps -ef | grep logprog          ← Kill the logprog process for the second time.  
# kill <pid>  
# cmviewcl -v                   ← The process should restart on the same node.  
  
# cmmmodpkg -v -e -n <primary_node> logprog  
  
# cmviewcl -v                   ← logprog now running on <adoptive node>.
```

- ## 8. Congratulations on configuring your first Serviceguard package!

## 7-19. SOLUTION: Check Your Understanding

### Directions

Answer the following questions.

1. Where should the application executables reside? A single copy on the shared disks along with the application database and other dynamic data files; or two copies, one on each server in a non-shared (private) volume group?

**Answer:**

Two copies: one on each server.

2. Should the shared file systems have entries in /etc/fstab? Why or why not?

**Answer:**

No. We want Serviceguard to mount and umount these file systems as packages are started and stopped on each node.

3. Should the shared volume groups be activated when the system boots (which happens by default)? How can this be prevented?

**Answer:**

No. We want Serviceguard to activate volume groups as needed. Edit /etc/lvmrc and set AUTO\_VG\_ACTIVATE=0.

4. How can you view the contents of the shared data with the cluster down? With the package down? In other words, how do you gain access to the database to do database maintenance, etc., and insure no users are accessing the application (either by taking down the package or the cluster)?

**Answer:**

```
# vgchange -c n vg01
# vgchange -a y vg01
# mount ...
```

5. If we kill an application on **node1** it automatically restarts on **node2**, but when we kill it on **node2** it does not automatically restart on **node1**. Why not? What must we do so it can switch back?

**Answer:**

Packages are not allowed to fail back until the administrator can correct the problem on **node1** that caused the original failover.

To switch the package back:

```
# cmmodpkg -e -n node1 pkg
```

6. How can we manually move a single package from **node1** to **node2**?

**Answer:**

```
<Node1> # cmhaltpkg -v pkg_name  
  
<Node1> # cmrunpkg -v -n <node2> pkg_name
```

7. How can we take one node out of service (and move applications/packages to adoptive nodes)? Return it to service? Will packages automatically return to primary node?

**Answer:**

```
<Node1> # cmhaltnode -v -f <node2>           ← to halt  
<Node1> # cmrunnode -v <node2>                ← to restart the node
```

No, packages will not automatically return to their node.

8. How can we stop one application, for example, for maintenance? How do we restart it?

**Answer:**

```
# cmhaltpkg -v pkg_name  
# cmrunpkg -v pkg_name
```

9. How does Serviceguard detect node failures, network failures, and application failures?

**Answer:**

Node failure — no heartbeat. We can miss heartbeats for many reasons other than node failure ... hung daemons, bad network, network traffic, real time user processes, etc.

Application failure — by definition, when the service monitor exits, the application has failed. An operator killing a process by mistake, a bug in the monitor script, etc. can mislead Serviceguard.

Network failure - all the network cards on the same subnet (primary and failover cards) attempt to communicate using their hardware addresses. (By default, this communication occurs every two seconds).

10. Can we perform database maintenance with the cluster down?

**Answer:**

Yes. See question 4 above.

Module 7  
**Packages and Services**

11. Why does Serviceguard require all nodes to be present when the cluster is first formed? How can we override this? Would we want to?

**Answer:**

If we are forming a cluster and one or more nodes is unreachable, there is always a possibility that the applications are already running on those unreachable nodes. In order to insure having the same application running on more than one node at a time does not corrupt our application database, the cluster does not form without all nodes present and accounted for. If we had an extended power outage, and for whatever reason one of the nodes is not booting (power supply destroyed by power surge), we would want to override this with the **-n** option of **cmrunc1**.

12. Under what circumstances is a floating IP address beneficial/necessary? Would this IP address be added to your DNS server? What host name should be used? Can an application/package have multiple IP addresses? Would this be of value?

**Answer:**

In order for clients to be able to transparently reconnect to their server processes after a failover, the application must have a floating IP address. Multiple IP addresses could be used to balance client traffic (and clients) over multiple subnets. The floating IP address would be assigned/maintained just like any other IP address. It should be documented on the DNS server. The hostname should be unique and reflect the application/package name.

13. How would you disable lower priority applications during a failover period? Notify administrators?

**Answer:**

The **start** script can check to see if the package is running on the adoptive system, and if so, can page the administrator; kill non-essential applications, and issue **cmhalt pkg** on non-essential packages, etc.

14. If application, server, disk, router, SCSI card, network card, etc fails, it should be transparent to the user. What if it is also transparent to the administrator? How can we insure against this?

**Answer:**

Eventually a redundant component will also fail and the package or cluster will be down. Ongoing monitoring is an essential component of high availability.

---

## 7-20. LAB Solution: Package Configuration #1: logprog Package

### Directions

In this lab, we will configure a package called `logprog` to run in our cluster.

---

**NOTE:** We will add additional features to this package in later modules.

---

### Part 1: Compile our application.

#### On Node1

1. The source code (`logprog.c`) for your application can be found in the `/labs` directory. Compile and save it as `/usr/lbin/logprog`. This will be the application around which we will build our package.

When this program runs, it will echo a status message to `/tmp/logprog.out` once every 3 seconds.

Compile your program as follows:

```
# cd /labs; cc logprog.c -o /usr/lbin/logprog
```

2. In one window, execute your `logprog` program. Open another window and verify the output with `tail -f /tmp/logprog.out`

```
Window1# /usr/lbin/logprog
Window2# tail -f /tmp/logprog.out
```

3. After verifying that `logprog` operates as expected, press `<ctrl>C` to stop it.

## Part 2: Create and Edit the Package Configuration File

### **On Node1**

1. Make a directory for the logprog package files.

#### **Answer:**

```
# cd /etc/cmcluster  
# mkdir logprog
```

2. Change directory to the logprog package directory that was just created. Once there, create the package configuration file template.

#### **Answer:**

```
# cd logprog  
# cmmakepkg -v -p logprog.conf
```

3. Next, edit the package configuration file template.

```
# vi logprog.conf
```

and modify the following parameters:

PACKAGE_NAME	logprog
NODE_NAME	<node1>
NODE_NAME	<node2>
RUN_SCRIPT	/etc/cmcluster/logprog/logprog.cntl
RUN_SCRIPT_TIMEOUT	NO_TIMEOUT
HALT_SCRIPT	/etc/cmcluster/logprog/logprog.cntl
HALT_SCRIPT_TIMEOUT	NO_TIMEOUT
SERVICE_NAME	logprog_service
SUBNET	x.x.x.x    ← Use netstat -in to obtain the 10-net subnet

4. Save the changes and exit the file.

## **Part 3: Create and Edit the Package Control Script, and Distribute this Script to All Nodes**

### **On Node1**

1. First, we create the package control script template.

**Answer:**

```
# cd /etc/cmcluster/logprog  
# cmmakepkg -v -s logprog.cntl
```

2. Secondly, edit the package control script template. Remember that this file is a script and that all variable assignments cannot contain any spaces.

```
# vi logprog.cntl
```

and modify the following parameters:

```
SERVICE_NAME[ 0 ]="logprog_service"  
SERVICE_CMD[ 0 ]="/usr/lbin/logprog"  
SERVICE_RESTART[ 0 ]="-r 2"
```

3. Save changes and exit the file.

4. Next, on Node2, create a subdirectory for your logprog package.

**Answer:**

```
# mkdir /etc/cmcluster/logprog
```

5. Now, back on Node1, make the logprog control script executable, and copy it to Node2 within the cluster. Also, copy your new /usr/lbin/logprog program to the same directory on Node2.

**Answer:**

```
# chmod 755 logprog.cntl  
# rcp logprog.cntl <Node2>:/etc/cmcluster/logprog/.
```

6. Copy your new /usr/lbin/logprog program to the same directory on Node2.

**Answer:**

```
# rcp /usr/lbin/logprog <Node2>:/usr/lbin/.
```

## **Part 4: Create and Distribute the Binary File, and Start the Package**

### **On Node1**

1. Check the package configuration file for errors and fix as needed.

**Answer:**

```
# cd /etc/cmcluster/logprog
# cmcheckconf -v -P logprog.conf
```

Fix any errors.

2. Since the configuration files are okay, apply the new configuration, thus compiling and distributing a new cluster binary file.

**Answer:**

```
# cd /etc/cmcluster/logprog
# cmaplyconf -v -P logprog.conf
```

3. View the package status.

**Answer:**

```
# cmviewcl -v
# cmviewcl -vp logprog
```

The package should be halted, but enabled to run on either node.

4. Start the package.

**Answer:**

```
# cmmodpkg -e logprog
```

5. We can test the `logprog` package behavior by doing the following:

```
# cmhaltpkg -v logprog  
# cmviewcl -vp logprog  
  
# cmrunkpkg -v -n <adoptive_node> logprog  
# cmmodpkg -v -e logprog  
  
# cmviewcl -vp logprog
```

6. We can further test our package by killing the logprog PID. We will kill the logprog process PID 3 times; the first two times because we specified 2 retries, and the third time to force a failover.

In the previous step, we moved the package to the `<adoptive_node>`. Execute the following steps on `<adoptive_node>` to force a failover back to `<primary_node>`.

```
# ps -ef | grep logprog           ← Kill the logprog process for the first time.  
# kill <pid>  
  
# ps -ef | grep logprog           ← Kill the logprog process for the second time.  
# kill <pid>  
  
# ps -ef | grep logprog           ← Kill the logprog process for the third time.  
# kill <pid>  
  
# cmviewcl -v                    ← The logprog package restarts on <primary node>.  
  
# cmmmodpkg -v -e -n <adoptive_node> logprog  
  
# cmviewcl -v                    ← logprog is now running on <primary node>.
```

7. The package should now be running on <primary\_node>. Execute the following steps on <primary\_node> to force a failover back to <adoptive\_node>.

```
# ps -ef | grep logprog          ← Kill the logprog process for the first time.  
# kill <pid>  
# ps -ef | grep logprog          ← The process should restart on the same node.  
# kill <pid>  
  
# ps -ef | grep logprog          ← Kill the logprog process for the second time.  
# kill <pid>  
# cmviewcl -v                   ← The process should restart on the same node.  
  
# cmmmodpkg -v -e -n <primary_node> logprog  
  
# cmviewcl -v                   ← logprog now running on <adoptive node>.
```

**Module 7**  
**Packages and Services**

8. Congratulations on configuring your first Serviceguard package!

---

## **Module 8 — Package Policies**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- Use FAILOVER\_POLICY to determine which node a package starts up on after the package fails.
- Use the FAILBACK\_POLICY to determine which node a package starts up on after a node fails.
- Determine proper Access Control Policies for a package
- Determine when NODE\_FAIL\_FAST or SERVICE\_FAIL\_FAST should be enabled.

## **8-1. SLIDE: Package Policies**

### **Package Policies**



- Failover policy
- Fallback policy
- Node fail fast
- Service fail fast

## **Student Notes**

### **Failover Policy**

The failover policy determines to which system a package fails during a failover situation. The two choices are to fail to a pre-configured system, or to fail to a system with the least number of packages.

### **Fallback Policy**

The fallback policy determines when a package returns to its original node. The two choices are to automatically return the package as soon as the original node is available, or to require the system administrator to manually return the package to the original node.

### **Node Fail Fast**

The NODE\_FAIL\_FAST parameter determines if the node is to fail upon a package failure. If multiple packages exist on one node and one of the packages fails, by default, just the failed package will move to another node. The node containing the failed package does not "fail" or TOC, and the remaining packages are allowed to continue on the node. This behavior is referred to as **node fail fast disabled**.

When a package is configured with "NODE\_FAIL\_FAST\_ENABLED YES", then the node will TOC if the package fails for any reason.

### **Service Fail Fast**

Along the same line of failing (TOCing) the node when a package fails, it is also possible to fail the node when a particular service process fails. This is generally considered drastic behavior; however, if the service process affects the health of the whole system, then such action may be appropriate. The default for every service process is **service fail fast disabled**.

## 8-2. SLIDE: Failover Policies

### Failover Policies



- Failover Policy is a per package policy.
- Used to select an adoptive node whenever the package is started.
- CONFIGURED\_NODE. Select the nodes in priority order. This option is the default.
- MIN\_PACKAGE\_NODE. Select the node which is running the least number of packages at this time.

### Student Notes

Configuration of a FAILOVER\_POLICY is done through the ASCII package configuration file. An excerpt of the description from the package file is shown below:

```
# Enter the failover policy for this package. This policy will be used
# to select an adoptive node whenever the package needs to be started.
# The default policy unless otherwise specified is CONFIGURED_NODE.
# This policy will select nodes in priority order from the list of
# NODE_NAME entries specified below.
#
# The alternative policy is MIN_PACKAGE_NODE. This policy will select
# the node, from the list of NODE_NAME entries below, which is
# running the least number of packages at the time this package needs
# to start.
```

FAILOVER\_POLICY

CONFIGURED\_NODE

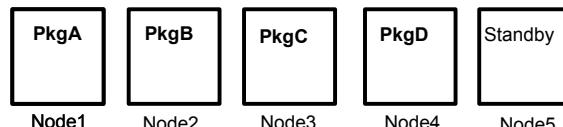
### 8-3. SLIDE: FAILOVER\_POLICY: CONFIGURED\_NODE

## FAILOVER\_POLICY: CONFIGURED\_NODE

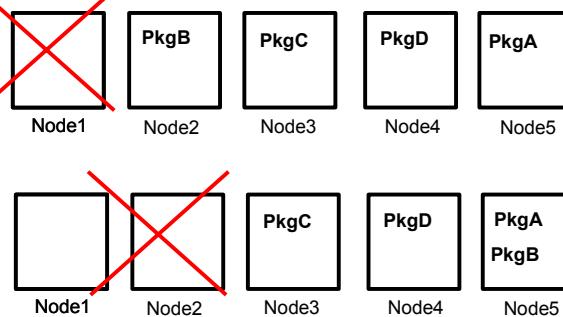


- Each package has a prioritized list of nodes.
- At failover time, package moves to next node in the list.

This is the default behavior.



- PkgA:  
Node1,Node5,Node2,Node3,Node4
- PkgB:  
Node2,Node5,Node1,Node3,Node4
- PkgC:  
Node3,Node5,Node2,Node4,Node1
- PkgD:  
Node4,Node5,Node3,Node1,Node2



### Student Notes

Packages begin their execution on their primary nodes (the first node listed in the package configuration file under the `NODE_NAME` parameter).

In the example on the slide, PkgA starts on Node1. After Node1 fails, PkgA moves to the next `NODE_NAME` in its prioritized list. For PkgA, this is Node5. Later, Node1 becomes healthy again and rejoins the cluster, but PkgA does not get returned to Node1.

Node2 then fails. PkgB moves to the next `NODE_NAME` in its prioritized list which is also Node5! In this situation, we would have two packages on Node5 and none on Node1.

A solution discussing how this situation could have been avoided is presented on the next slide.

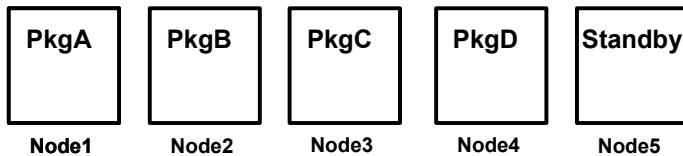
The `CONFIGURED_NODE` configuration is useful when you want to control specifically what the standby will be for a particular package or you simply want to connect the shared disks between only 2 nodes.

## 8-4. SLIDE: FAILOVER\_POLICY: MIN\_PACKAGE\_NODE

### FAILOVER\_POLICY: MIN\_PACKAGE\_NODE



- At package start time, package will be started on the node with the fewest packages.
- Package will start only on nodes listed in the NODE\_LIST.



### Student Notes

The failover algorithm for MIN\_PACKAGE\_NODE is:

- At the time of the failure event, the failing package determines the list of possible adoptive nodes by querying the current state of the nodes in its configured list.
- The adoptive node is selected by selecting the node that is running fewer packages.
- If multiple nodes have the same number of packages, then the node listed first in the NODE\_NAME list is selected.

## 8-5. SLIDE: Rotating Standby

### Rotating Standby



Failover Policy can be used to perform the rotating standby operation.

Example:

	PkgA	PkgB	PkgC	PkgD	Standby
1					
2					
3					
4					

The diagram illustrates a four-step process of package distribution across five nodes (Node1 to Node5). In step 1, packages PkgA through PkgD are assigned to Node1, Node2, Node3, and Node4 respectively, while Node5 is designated as the Standby. In step 2, Node1 fails, so PkgA moves to Node5. In step 3, Node1 rejoins the cluster, becoming the logical standby. In step 4, another failure occurs on Node2, causing PkgB to move to Node1. The red X marks indicate failed nodes, and the arrows show the movement of packages between nodes.

### Student Notes

The above slide walks through an example of a "rotating standby" which can easily be implemented with the FAILOVER\_POLICY of MIN\_PACKAGE\_NODE.

1. In the example, all packages can run on all nodes, and the data for each package is connected to all nodes in the cluster.
2. When node1 fails, PkgA moves to node5 because it has the fewest number of packages.
3. When node1 rejoins the cluster, node1 becomes the **logical standby**, because if a node were to fail, node1 would have the fewest number of packages running on it.

As PkgA continues on node5, another failure occurs, this time on node2. PkgB moves to node1, because it has the least number of packages. After node2 is repaired and rejoins the cluster, node2 will become the logical standby.

## 8-6. SLIDE: Failback Policies

### Fallback Policies



- **FAILBACK\_POLICY** is a per package policy.
  - Used to allow the movement of a package back to its primary node.
  - MANUAL. No failback is attempted. Packages must manually be returned to the primary node when desired, or will be moved back if a failure on the current node occurs. This is the default.
  - AUTOMATIC. Packages will be moved from a non-primary node back to the primary node whenever the primary node has the ability to run the package.

### Student Notes

Configuration of FAILBACK\_POLICY is done through the ASCII package configuration file. An excerpt of the description from the package configuration file is shown below:

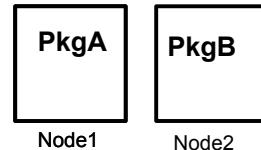
```
# Enter the failback policy for this package. This policy will be
# used to determine what action to take when a package is not
# running on its primary node and its primary node is capable of
# running the package. The default policy unless otherwise specified
# is MANUAL. The MANUAL policy means no attempt will be made to
# move the package back to its primary node when it is running on an
# adoptive node.
#
# The alternative policy is AUTOMATIC. This policy will attempt to
# move the package back to its primary node whenever the primary
# node is capable of running the package.
```

## 8-7. SLIDE: Example of Automatic Failback

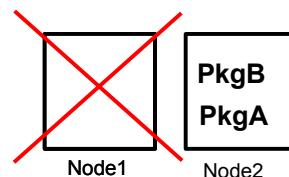
### Example of Automatic Failback



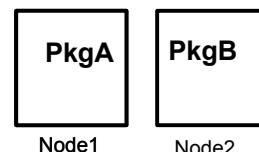
- PkgA and PkgB configured for node1 and node2.



- Node1 panics. PkgA moves to Node2



- Node1 reboots and PkgA is automatically halted and started on node1.



### Student Notes

In the above slide, assume `PkgA` and `PkgB` have their `FAILBACK_POLICY` set to `AUTOMATIC`.

When `node1` goes down, the normal failover activities take place, moving the package to `node2`. When `node1` reboots and rejoins the cluster, the `FAILBACK_POLICY` is referenced to determine whether to return `PkgA` to its original node. Since the policy is set to `AUTOMATIC`, when `node1` rejoins the cluster, `PkgA` is halted automatically on `node2`, and restarted on `node1`.

## 8-8. SLIDE: Access Control Policies

### Access Control Policies



- **Role-Based Access Control**

By default, only the root user can access/control a package  
Other users (optionally from other hosts) can be allowed certain privileges

- **Access Control Policy Parameters**

USER_NAME	ANY_USER –or- a list of specific users
USER_HOST	Where the user can execute SG commands
USER_ROLE	PACKAGE_ADMIN

- **Limitations**

Only the root user can configure a cluster or packages

### Student Notes

Beginning with Serviceguard A.11.16, and on a per-package basis, there are options in the package configuration file that allow non-root users to manage/monitor a package. These options follow:

USER\_NAME  
USER\_HOST  
USER\_ROLE

The first line must be USER\_NAME, second USER\_HOST, and third USER\_ROLE.

1. USER\_NAME can either be ANY\_USER, or a maximum of 8 login names from the /etc/passwd file on user host.
2. USER\_HOST is where the user can issue package-specific Serviceguard commands. If using Serviceguard Manager, it is the COM (Cluster Object Manager) server. Choose one of these three values: ANY\_SERVICEGUARD\_NODE, or (any) CLUSTER\_MEMBER\_NODE, or a specific node. For node, use the official hostname from domain name server, and not an IP address or fully qualified name.

3. `USER_ROLE` may only be `PACKAGE_ADMIN`, which includes MONITOR, plus administrative commands for packages in the cluster

Access control policy does not set a role for configuration capability. To configure a cluster or a package, a user must log on to one of the cluster's nodes as root (UID=0). Access control policy cannot limit root users' access.

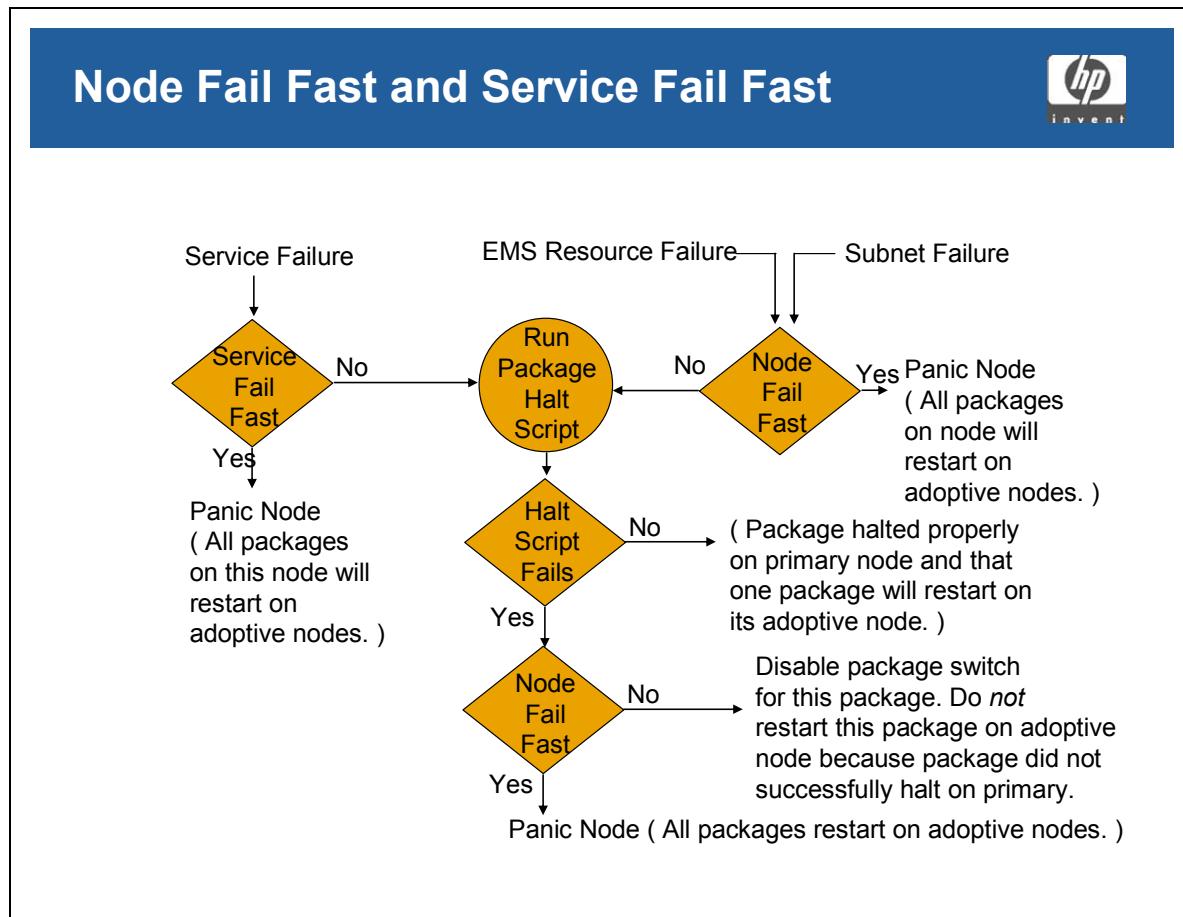
`MONITOR` and `FULL_ADMIN` can only be set in the cluster configuration file, and they apply to the entire cluster. `PACKAGE_ADMIN` can be set in the cluster or a package configuration file. If set in the cluster configuration file, `PACKAGE_ADMIN` applies to all configured packages. If set in a package configuration file, `PACKAGE_ADMIN` applies to that package only.

Conflicting or redundant policies will cause an error while applying the configuration, and will stop the process. If a role is configured in a cluster, do not duplicate it in a package.

Example: to configure a role for user `oracle` from node `rx26u891` to administer the `oracle` package, enter the following in the `oracle` package config file:

```
USER_NAME oracle
USER_HOST rx26u891
USER_ROLE FULL_ADMIN
```

## 8-9. SLIDE: Node Fail Fast and Service Fail Fast



### Student Notes

#### Node Fail Fast

If multiple packages exist on one node and one of the packages fails, by default, only the failed package will move to another node. The node containing the failed package does not panic, and the remaining packages are allowed to continue running on the node. This behavior is the default behavior and is configured with "NODE\_FAIL\_FAST\_ENABLED NO" in the package configuration file.

When a package is configured with "NODE\_FAIL\_FAST\_ENABLED YES", the node will fail, or TOC if:

- The node loses contact with the subnet configured by its SUBNET parameter,
- The package halt script fails, or
- A Resource failure occurs (from EMS: see Module 16).

Failing a node does not give the software a chance to gracefully bring down other packages or non-Serviceguard software, so this parameter should only be set when a package contains a service that is so mission-critical that loss of that service makes the node unusable.

This behavior is referred to as **node fail fast enabled**.

### Service Fail Fast

Similarly to Node Fail Fast Enabled, it is also possible to fail the node when a particular service process fails. This is generally considered drastic behavior. However, if the service process affects the health of the whole system, then such action may be appropriate.

The default for every service process is to disable this attribute. This attribute can be selectively enabled on a per-service process basis. The variable `SERVICE_FAIL_FAST_ENABLED` is set to YES or NO to achieve the desired result.

The slide shows the flow of what happens when a service fails or a package fails depending on the two “fail fast” settings.

### Service Failures

When a service fails, the default behavior is to restart the service if any restarts exist. If all the restarts have been used (or none were configured), then the package halt script is executed to bring down the package. If, however, service fail fast is enabled, the node panics immediately and the halt script is not executed.

### Package Failures

Three situations can cause a package to fail on a healthy node:

- Subnet failure. If the package is configured to be dependent on a subnet and that subnet is determined to be inaccessible by the local system, then a package failure will occur.
- Halt script failure. If an error is encountered or the halt script times out during the execution of the halt script, the package has failed.
- If EMS is being used (to be discussed later in Module 16), and if there is an EMS resource failure, the package will be brought down and potentially failed over to an adoptive node.

### Example

A database package is configured with “`SERVICE_FAIL_FAST_ENABLED NO`” and “`NODE_FAIL_FAST_ENABLED YES`”.

If a critical service process prematurely terminates, the halt script for the database package is executed. If the halt script succeeds in gracefully bringing down the database package, then the node is disabled for the package and the package restarts on an adoptive node in the cluster. A node failure does not occur.

If, however, the halt script fails (assume there were users logged into the database and it could not be shutdown), the halt script error would cause a node failure (panic), since “`NODE_FAIL_FAST_ENABLED YES`” was set. The panic of the node causes the `cmlvmd` process to disappear, and the package IP address is no longer configured on the local LAN card.

**Module 8**  
**Package Policies**

If "SERVICE\_FAIL\_FAST\_ENABLED NO", the halt script would have failed and the node would have been left up. This would have left the cm1vmd process running and, depending on where and how the halt script failed, would have left the package IP address configured. The package would not be able to start on another node due to the volume group being exclusively accessed by the first node and the IP address still assigned.

---

## 8-10. LAB: FAILBACK POLICY

### Directions

1. Modify your `logprog` package to fail back when the primary node comes back to the cluster, and test it out.

**Answer:**

In the "`logprog.conf`" file, change the line that now reads ...

FAILBACK_POLICY	MANUAL
-----------------	--------

to read ...

FAILBACK_POLICY	AUTOMATIC
-----------------	-----------

and then reconfigure the package in the normal way.

```
# cmhaltcl -f
# cd /etc/cmcluster/logprog
# cmcheckconf -v -P logprog
# cmapplyconf -P logprog.conf
# cmrunc1
# cmviewc1 -vp logprog
# cmhaltnode -f (on primary)
# cmrunnode (on primary)
```

2. Did the `logprog` package come back to the primary node?

Module 8  
**Package Policies**

3. While there might be compelling reasons to use the AUTOMATIC fallback policy, we will be performing other operations with the logprog package that require the MANUAL fallback policy. Therefore, modify your logprog package configuration file again to reflect a MANUAL fallback. Then check and apply.

## **8-11. LAB: Access Control Policy**

# Directions

In this lab, we will give user50 PACKAGE\_ADMIN status to the logprog package, and we'll give this user this capability only on nodes within your cluster. While we could have given PACKAGE\_ADMIN capability to individual users in the cluster ASCII file, there is no way to prevent those users from executing commands on packages for which they would not ordinarily be responsible.

In the package configuration file, assigning individual users `PACKAGE_ADMIN` capability will allow us to more tightly control who manages which packages.

1. We don't currently have a user50 on our systems, so let's add one. Assign this user a password of "hp". Be sure to do this on both nodes in your cluster.
  2. Edit the logprog package configuration file and give user50 the full access to this package from either node in your cluster.
  3. Check and apply the logprog package configuration file.
  4. Login as user50 and halt the logprog package. Since the /usr/sbin directory (where most of the Serviceguard commands reside) is not in user50's PATH, you will need to specify the absolute path to the commands below. Start the package again on the adoptive node. Then bring it back to its primary node.

Module 8  
**Package Policies**

5. Login as root again for the remainder of the lab.

---

## 8-12. LAB: SERVICE\_FAIL\_FAST\_ENABLED (Optional)

### Directions

#### Special Note

---

This lab and the following lab provide an opportunity to use the SERVICE\_FAIL\_FAST\_ENABLED and NODE\_FAIL\_FAST\_ENABLED options on packages. These options are often not well understood.

These options generally *should not* be used in production environments.

Both of these options will cause a server to panic when an application-related fault occurs. Specifically, with SERVICE\_FAIL\_FAST\_ENABLED, a service (application) failure will cause that node to panic. Generally, application failure can be handled with less severe action than a node panic. Also, it is entirely possible that when the package restarts on the adoptive node it will also fail there, causing both nodes to panic. For these reasons, SERVICE\_FAIL\_FAST\_ENABLED is virtually never implemented in production environments.

The NODE\_FAIL\_FAST\_ENABLED option causes a node to panic if the package attempts to halt, and the halt script fails. Again, in a production environment, this would generally be avoided. Less severe methods than a panic can be used to overcome an application-related failure (in this case a halt script failure). Also, the error could be encountered on the adoptive node, causing both nodes to panic.

1. First, halt the cluster. Then, restart the cluster. Finally, make sure your package is running.
  
2. Before continuing, make a backup copy of the `logprog.conf` and the `logprog.cntl` files.

## Module 8

# Package Policies

3. Now, modify the ASCII configuration file for the `logprog` package so that `SERVICE_FAIL_FAST_ENABLED` is enabled.
  4. Check the modified `logprog` configuration file. Make any necessary corrections and recheck if necessary. Apply the modified configuration. Will Serviceguard allow you to make these changes while the cluster is active?
  5. Start the `logprog` package. Kill it until a failover occurs. What happened? After a minute or so, the node on which `logprog` is running will panic.

## 8-13. LAB: NODE\_FAIL\_FAST\_ENABLED (Optional)

## Directions

In this part of the lab, we will configure and test the package again: this time by enabling the `NODE_FAIL_FAST_ENABLED` policy.

1. Wait for the failed system to reboot and rejoin the cluster.
  2. Modify the ASCII configuration file for the `logprog` package so that `SERVICE_FAIL_FAST_ENABLED` is disabled, and `NODE_FAIL_FAST_ENABLED` is enabled.
  3. Halt the `logprog` package. Check the modified configuration. If necessary, correct any errors and recheck the configuration. Apply the modified configuration. Verify that `SERVICE_FAIL_FAST_ENABLED` is disabled, and that `NODE_FAIL_FAST_ENABLED` is enabled.
  4. Start the `logprog` package. Ensure that package switching is enabled.
  5. Kill the `logprog` until a failover occurs. Did the node fail? Why or why not?

Module 8  
**Package Policies**

6. Re-enable the package switching for the node that just failed.
  
  
  
  
  
  
7. On the node where `logprog` is currently running, make a backup copy of the control script for `logprog`, if you have not already done so.
  
  
  
  
  
  
8. On the node where `logprog` is currently running, force an error in the `halt` script for `logprog`. You can do this by editing the `logprog.cntl` file. Add the command `exit 99` to the `customer_defined_halt_cmds` function.

```
# vi logprog.cntl

function customer_defined_halt_commands
{
    exit 99
}
```
  
  
  
  
  
  
9. View the `node_switching` parameters for the `logprog` package. Make sure that `logprog` is enabled on all nodes. If not, enable it.
  
  
  
  
  
  
10. Kill the `logprog` enough times to force a failover. Did the node panic this time? Why?

11. Once the failed system reboots, restore the backup copy of the `logprog` control script.
  12. Halt the `logprog` package if it is running. Modify the ASCII configuration file for `logprog` to disable both `NODE_FAIL_FAST_ENABLED` and `SERVICE_FAIL_FAST_ENABLED`. Check and apply the modified configuration.
  13. Finally, start the `logprog` package.

## **8-14. LAB Solution: FAILBACK POLICY**

### **Directions**

1. Modify your `logprog` package to fail back when the primary node comes back to the cluster, and test it out.

#### **Answer:**

In the "`logprog.conf`" file, change the line that now reads ...

FAILBACK_POLICY	MANUAL
-----------------	--------

to read ...

FAILBACK_POLICY	AUTOMATIC
-----------------	-----------

and then reconfigure the package in the normal way.

```
# cmhaltcl -f
# cd /etc/cmcluster/logprog
# cmcheckconf -v -P logprog
# cmapplyconf -P logprog.conf
# cmrunc1
# cmviewcl -vp logprog
# cmhaltnode -f (on primary)
# cmrunnode      (on primary)
```

2. Did the `logprog` package come back to the primary node?

#### **Answer:**

Yes.

3. While there might be compelling reasons to use the `AUTOMATIC` fallback policy, we will be performing other operations with the `logprog` package that require the `MANUAL` fallback policy. Therefore, modify your `logprog` package configuration file again to reflect a `MANUAL` fallback.

### **Answer:**

```
# cmhaltpkg logprog  
# vi /etc/cmcluster/logprog.conf
```

Change FAILBACK from AUTOMATIC to MANUAL

```
# cmcheckconf -P /etc/cmcluster/logprog.conf  
# cmapplyconf -P /etc/cmcluster/logprog.conf
```

## **8-15. LAB Solution: Access Control Policy**

### **Directions**

In this lab, we will give `user50 PACKAGE_ADMIN` status to the `logprog` package, and we'll give this user this capability only on nodes within your cluster. While we could have given `PACKAGE_ADMIN` capability to individual users in the cluster ASCII file, there is no way to prevent those users from executing commands on packages for which they would not ordinarily be responsible.

In the package configuration file, assigning individual users `PACKAGE_ADMIN` capability will allow us to more tightly control who manages which packages.

1. We don't currently have a `user50` on our systems, so let's add one. Assign this user a password of "hp". Be sure to do this on both nodes in your cluster.

**Answer:**

```
<node1> # useradd -m user50
<node1> # passwd user50
<node2> # useradd -m user50
<node2> # passwd user50
```

2. Edit the `logprog` package configuration file and give `user50` the full access to this package from either node in your cluster.

**Answer:**

```
# vi /etc/cmcluster/logprog/logprog.conf
```

USER_NAME	user50
USER_HOST	CLUSTER_MEMBER_NODE
USER_ROLE	PACKAGE_ADMIN

3. Check and apply the `logprog` package configuration file.

**Answer:**

```
# cmcheckconf -P /etc/cmcluster/logprog.conf
# cmaplyconf -P /etc/cmcluster/logprog.conf
```

4. Login as `user50` and halt the `logprog` package. Since the `/usr/sbin` directory (where most of the Serviceguard commands reside) is not in `user50`'s `PATH`, you will need to specify the absolute path to the commands below. Start the package again on the adoptive node. Then bring it back to its primary node.

## Answer:

```
# login user50
$ /usr/sbin/cmhaltpkg logprog
$ /usr/sbin/cmrunkpkg -n <node2> logprog
$ /usr/sbin/cmviewcl -vp logprog ← and verify that it's running on <node2>
$ /usr/sbin/cmhaltpkg logprog
$ /usr/sbin/cmmodpkg -e logprog
$ /usr/sbin/cmviewcl -vp logprog ← and verify that it's running on <node1>
```

5. Login as root again for the remainder of the lab.

## Answer:

```
# login root
```

## **8-16. LAB Solution: SERVICE\_FAIL\_FAST\_ENABLED (Optional)**

### **Directions**

#### **Special Note**

This lab and the following lab provide an opportunity to use the SERVICE\_FAIL\_FAST\_ENABLED and NODE\_FAIL\_FAST\_ENABLED options on packages. These options are often not well understood.

These options generally *should not* be used in production environments.

Both of these options will cause a server to panic when an application-related fault occurs. Specifically, with SERVICE\_FAIL\_FAST\_ENABLED, a service (application) failure will cause that node to panic. Generally, application failure can be handled with less severe action than a node panic. Also, it is entirely possible that when the package restarts on the adoptive node it will also fail there, causing both nodes to panic. For these reasons, SERVICE\_FAIL\_FAST\_ENABLED is virtually never implemented in production environments.

The NODE\_FAIL\_FAST\_ENABLED option causes a node to panic if the package attempts to halt, and the halt script fails. Again, in a production environment, this would generally be avoided. Less severe methods than a panic can be used to overcome an application-related failure (in this case a halt script failure). Also, the error could be encountered on the adoptive node, causing both nodes to panic.

1. First, halt the cluster. Then, restart the cluster. Finally, make sure your package is running.

**Answer:**

```
# cmhaltcl -v -f  
# cmrunc1 -v
```

2. Before continuing, make a backup copy of the logprog.conf and the logprog.cntl files.

**Answer:**

```
# cd /etc/cmcluster/logprog  
# cp logprog.conf logprog.conf.save  
# cp logprog.cntl logprog.cntl.save
```

3. Now, modify the ASCII configuration file for the `logprog` package so that `SERVICE_FAIL_FAST_ENABLED` is enabled.

**Answer:**

```
# vi logprog.conf  
  
SERVICE_FAIL_FAST_ENABLED YES
```

4. Check the modified `logprog` configuration file. Make any necessary corrections and recheck if necessary. Apply the modified configuration. Will Serviceguard allow you to make these changes while the cluster is active?

**Answer:**

```
# cmcheckconf -v -P logprog.conf  
# cmhaltpkg -v logprog  
# cmapplyconf -v -P logprog.conf
```

5. Start the `logprog` package. Kill it until a failover occurs. What happened?

**Answer:**

```
# cmmodpkg -v -e logprog  
# ps -ef | grep logprog  
# kill <pid>
```

After a minute or so, the node on which `logprog` is running will panic.

## **8-17. LAB Solution: NODE\_FAIL\_FAST\_ENABLED (Optional)**

### **Directions**

In this part of the lab, we will configure and test the package again: this time by enabling the NODE\_FAIL\_FAST\_ENABLED policy.

1. Wait for the failed system to reboot and rejoin the cluster.
2. Modify the ASCII configuration file for the logprog package so that SERVICE\_FAIL\_FAST\_ENABLED is disabled, and NODE\_FAIL\_FAST\_ENABLED is enabled.

**Answer:**

```
# vi logprog.conf
```

SERVICE_FAIL_FAST_ENABLED	NO
NODE_FAIL_FAST_ENABLED	YES

3. Halt the logprog package. Check the modified configuration. If necessary, correct any errors and recheck the configuration. Apply the modified configuration. Verify that SERVICE\_FAIL\_FAST\_ENABLED is disabled, and that NODE\_FAIL\_FAST\_ENABLED is enabled.

**Answer:**

```
# cmhaltpkg -v logprog
# cmcheckconf -v -P logprog.conf
# cmaplyconf -v -P logprog.conf
```

4. Start the logprog package. Ensure that package switching is enabled.

**Answer:**

```
# cmmodpkg -v -e logprog
# cmviewcl -v -p logprog
```

5. Kill the `logprog` until a failover occurs. Did the node fail? Why or why not?

**Answer:**

```
# ps -ef | grep logprog
# kill <pid>
```

The node does not fail because the halt script was successful. This is the difference between `NODE_FAIL_FAST` and `SERVICE_FAIL_FAST`.

6. Re-enable the package switching for the node that just failed.

**Answer:**

```
# cmmodpkg -v -e -n node_name logprog
```

7. On the node where `logprog` is currently running, make a backup copy of the control script for `logprog`, if you have not already done so.

**Answer:**

```
# cp logprog.cntl logprog.cntl.save
```

8. On the node where `logprog` is currently running, force an error in the `halt` script for `logprog`. You can do this by editing the `logprog.cntl` file. Add the command `exit 99` to the `customer_defined_halt_cmds` function.

```
# vi logprog.cntl
```

```
function customer_defined_halt_commands
{
    exit 99
}
```

9. View the `node_switching` parameters for the `logprog` package. Make sure that `logprog` is enabled on all nodes. If not, enable it.

**Answer:**

```
# cmviewcl -vp logprog
# cmmodpkg -v -e -n node_name logprog
```

**Module 8**  
**Package Policies**

10. Kill the `logprog` enough times to force a failover. Did the node panic this time? Why?

**Answer:**

```
# ps -ef | grep logprog  
# kill <pid>
```

```
<etc>
```

The node fails this time because the halt script fails (with return code 99).

11. Once the failed system reboots, restore the backup copy of the `logprog` control script.

**Answer:**

```
# cd /etc/cmcluster/logprog  
# mv logprog.cntl.save logprog.cntl
```

12. Halt the `logprog` package if it is running. Modify the ASCII configuration file for `logprog` to disable both `NODE_FAIL_FAST_ENABLED` and `SERVICE_FAIL_FAST_ENABLED`. Check and apply the modified configuration.

**Answer:**

```
# cmhaltpkg -v logprog  
# mv logprog.conf.save logprog.conf  
# cmcheckconf -v -P logprog.conf  
# cmaplyconf -v -P logprog.conf  
# cmviewcl -v -p logprog
```

13. Finally, start the `logprog` package.

**Answer:**

```
# cmmodpkg -e logprog
```

---

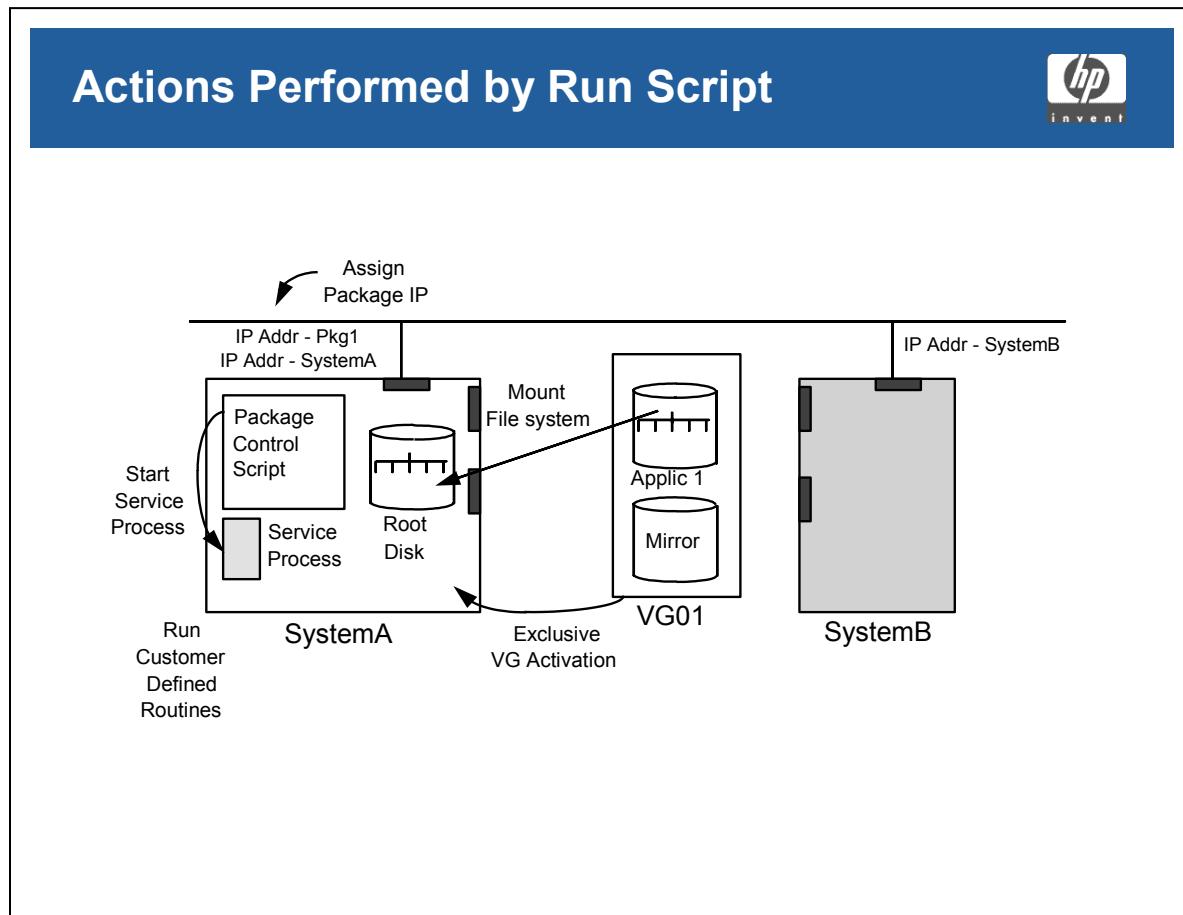
## **Module 9 — Application Monitoring Scripts**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- Configure and implement an application monitor.
- List actions performed by the package control script.
- List three rules for Serviceguard service processes.

## 9–1. SLIDE: Actions Performed by Run Script



### Student Notes

The package run script (also called the package control script) is responsible for starting the application on the nodes within the cluster.

The following actions are **always** performed by the package run script:

- Activating the volume group, if being used.
- Checking and mounting the file system, if being used.
- Assigning the package IP address, if being used.
- Executing any customer-defined run commands, if being used.
- Starting the service processes.

## 9-2. SLIDE: Control Script

### Control Script



```
if  [[ $1 = "start"  ]]
then
    activate_volume_group
    check_and_mount
    add_ip_address
    customer_defined_run_cmds
    start_services
elif [[ $1 = "stop"   ]]
then
    halt_services
    customer_defined_halt_cmds
    remove_ip_address
    umount_fs
    deactivate_volume_group
fi
```

Excerpt from Package Control Script

### Student Notes

The slide shows an excerpt from the package control script. The excerpt is an abbreviated version of the main program contained within the package control script.

The major activities performed by the control script are grouped into subroutines. The order in which the subroutines are called corresponds to the order in which the major activities are performed.

---

**NOTE:** The sequence of activities used to start the package is in reverse order to the sequence of activities used to stop the package.

---

### 9-3. SLIDE: Rules for Service Processes

## Rules for Service Processes



### For Serviceguard Version 11.17:

#### **General Rules:**

- A package may have zero to a maximum of 30 service processes.
- Each service process can have zero to an unlimited number of restarts.
- If a single service process cannot be restarted, the package is halted.

#### **A Service Process MUST:**

- Be defined in the package configuration file.
- Be invoked from within the control script.
- Always be present in order for the package to be considered healthy.

#### **A Service Process must NOT:**

- Be a process which is spawned and respawned as requests come in.
- Be a process which only the application can invoke ( i.e. Oracle daemons ).

## Student Notes

Care must be taken when defining service processes.

### **How Are Service Processes Started and Monitored by Serviceguard?**

Service processes are defined in both the package configuration file and the package control script. Within the package control script, the parameter SERVICE\_CMD contains the full path name of the program to run as the service process.

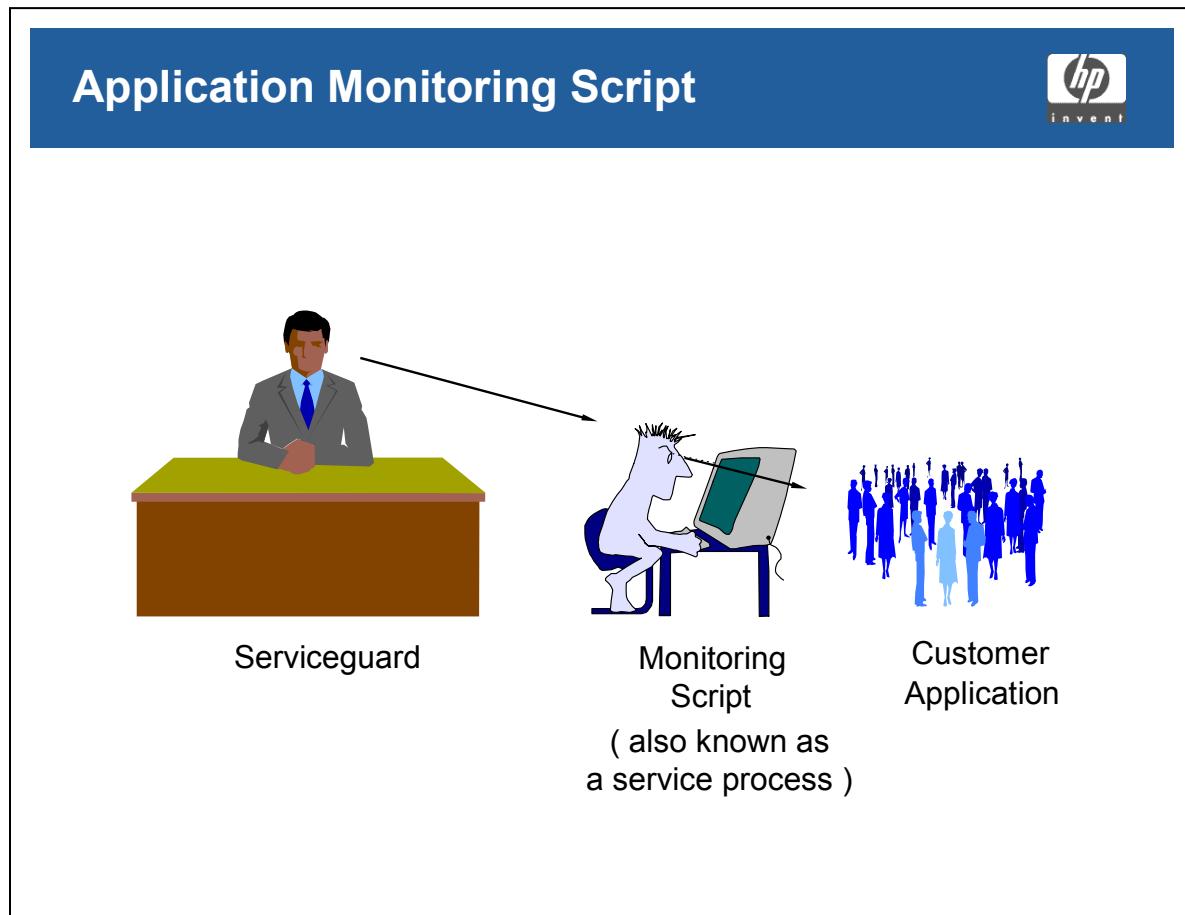
The service process specified in SERVICE\_CMD is spawned by the `cmrunserv` command. The resulting process ID is then monitored by Serviceguard to determine the health of the application.

Based upon the above description, the following do not make good service processes:

- **Launch program.** A launch program is typically a script which spawns ( or launches ) a number of other programs and background processes related to an application and then exits. An example of a launch program is `dbstart`, a database program which starts the Oracle background daemon processes and then terminates.

- **Connection-oriented processes.** Connection-oriented processes are those which are spawned when a user request comes in. These processes service the user request and then exit when the request is completed. Examples of connection-oriented processes are those spawned by `inetd`, such as `telnetd`, `ftpd`, and `logind`.
- **Application-spawned processes.** Application-spawned processes are those which are started by some other application process or program and cannot be easily spawned from the command line or through the `cmrunserv` command. Examples of application-spawned processes are the Oracle daemons, such as `dbwr` or `lgwr`. Another example would be daemons started by the operating system at boot time, such as `nfsd`. Because Serviceguard does not start these processes directly, it does not know which PID to monitor since there could be multiple occurrences of these processes.

## 9-4. SLIDE: Application Monitoring Script



### Student Notes

An application monitoring script is a persistent process that checks periodically to see if the application is still running. If the application is found to still be healthy, the script sleeps for a predefined number of seconds before checking again.

If, during one of its checks, it finds the application is not healthy, it may try to resolve the problem or simply exit.

Serviceguard uses these application monitoring programs as service processes started by the `cmrunserv` command. By using these monitoring scripts, Serviceguard is only concerned with whether the monitoring program is running. It leaves it up to the monitoring script to ensure the health of the application.

A major advantage of a monitoring script is the amount of flexibility that can be used to monitor the health of the application. In addition to checking whether critical application processes are running, the monitoring script can also check items like file systems being full, errors in log files, and deadlock conditions in database tables.

A second advantage of the monitoring scripts is the ability to build logic into the scripts to fix problems related to the application when they occur. Rather than exiting when an application process dies, the monitoring script could first try to respawn the application process before exiting. Rather than terminating when the file system becomes full, the monitoring script could try to free up space ( maybe by looking for core files and removing them ) before terminating.

The end result is that Serviceguard watches the monitoring script, and the monitoring script watches all aspects of the application.

The creation of monitoring scripts does not always have to be done from scratch. Many applications have sample scripts and instructions on how to "package" the application to run in a Serviceguard environment.

## 9-5. SLIDE: Example: The logprog Program

### Example: The logprog Program



- Activate volume group
  - File system check and mount
  - Add IP address
  - Customer defined run commands
- Start the `logprog` program here.

`logprog`

- Start `logprog` Monitoring Process (aka a service process)  
`logprog.mon` checks to see if the `logprog` program is running.
  - If daemons are OK, sleep for 10 seconds, and check again.
  - If daemons are NOT OK, try to fix. If cannot be fixed, exit.

### Student Notes

The slide shows the five major activities performed by the package control script when starting the `logprog` program with a monitoring script.

1. Activate the volume group if one exists. Please recall, though, that for the `logprog` package, there is no volume group currently being used.
2. Check and mount any file systems related to the package. Once again, however, for the `logprog` package, there are no file systems.
3. Add the IP address for the `logprog` package. Here also, we omit this step since the `logprog` package does not require IP addresses.
4. Execute any commands in the `customer_defined_run_cmds` function in the package control script. This function is where the command to start `logprog` is added.
5. Start the `logprog` monitoring script to monitor the `logprog` program started in step 4.

## 9–6. LAB: logprog Package Monitoring Script

### Directions

- Instead of using `/usr/lbin/logprog` as the service, create a separate service monitoring script called `/etc/cmcluster/logprog/logprog.mon` as follows:

```
# cd /etc/cmcluster/logprog

# vi logprog.mon

#!/usr/bin/sh

LOG="/etc/cmcluster/logprog/logprog.cntl.log"

echo "Now entering the logprog package monitor \c"      >> $LOG
echo "script on $(hostname) at $(date)."                  >> $LOG
while true
do
    if ps -ef | grep -v grep | grep -q "/usr/lbin/logprog"
        then
            echo "Package logprog apparently ok \c"
            echo "on $(hostname) at $(date)."
            sleep 10
        else
            echo "Package logprog failed at $(date) \c"
            echo "from node $(hostname)."
            exit
        fi
    done >> $LOG
```

The extra echo statements in the above script are sometimes referred to as "debug lines". Once the package and its scripts are working properly, these debug lines may be safely commented out. Good programming practice would suggest that these "debug lines" would be commented out (but not removed) from the script since they may be needed again in the future.

- Make the monitor script executable.
- Copy the monitor script to other system in your cluster.

Module 9  
**Application Monitoring Scripts**

4. In the `logprog` package control script `/etc/cmcluster/logprog/logprog.cntl`, change the following two lines. To accomplish this, remove the `SERVICE_CMD[0]="/usr/lbin/logprog"` line and replace it with the following. Also, please remember to change the number of restarts to 0.

```
SERVICE_CMD[0]="/etc/cmcluster/logprog/logprog.mon"
SERVICE_RESTART[0]="-r 0"
```

In other words, Serviceguard will now be watching the `logprog.mon` process, and we will use zero restarts. In turn, the `logprog.mon` process will be watching the health of the `logprog` package itself.

5. In the `customer_defined_run_cmds` function of the `logprog` package control script, `/etc/cmcluster/logprog/logprog.cntl`, remove the line beginning with the colon (`:`). Then, in its place, enter the command to start our `logprog`. Do not remove the `"test_return 51"` line.

```
/usr/lbin/logprog &
```

**Important:** Do not forget to put your `logprog` process in the background!

6. In the `customer_defined_halt_cmds` function, remove the line beginning with the colon (`:`), and also remove any blank lines. Do not remove the `"test_return 52"` line. Then, replace it (them) with the following few lines. This will stop our `logprog` if it is running:

```
function customer_defined_halt_cmds
{
LOG="/etc/cmcluster/logprog/logprog.cntl.log"

echo "Now entering the customer_defined_halt_cmds \c" >> $LOG
echo "on $(hostname) at $(date)." >> $LOG
if ps -ef | grep -v grep | grep -q "/usr/lbin/logprog"
    then
        echo "Found the logprog process at $(date)." >> $LOG
        echo "Killing logprog process at $(date)." >> $LOG
        kill -9 $(ps -ef | grep -v grep | grep "/usr/lbin/logprog" \
            | cut -c10-14)
    else
        echo "Note: In customer_defined_halt_cmds \c" >> $LOG
        echo "on $(hostname), and could not find the \c" >> $LOG
        echo "logprog process at $(date)." >> $LOG
    fi
```

Once again, the extra echo statements in the above code are sometimes referred to as "debug lines". Once the package and its scripts are working properly, these debug lines may be safely commented out but not removed (as discussed above in step 1).

**NOTE:** If Serviceguard believes the halt script did not properly stop the application, the halt script will fail, and Serviceguard will not start the package on the adoptive node to avoid risking application data corruption.

7. Copy the revised package control script to `node2`.
  8. Verify that all the pieces work. Open a terminal window and execute `tail -f /etc/cmcluster/logprog/logprog.cntl.log` to view the package activities.
  9. Halt the cluster and then verify that the `logprog` program has also exited.
  10. Now, restart the cluster, which automatically restarts the `logprog` package.
  11. Kill the `logprog` process ( one time ) and verify it restarts on the adoptive node.
  12. Verify that the echo message was written to the package log file on the failing node.
  13. Return the `logprog` package to the primary node by halting, then immediately restarting the cluster.
  14. Congratulations on building and testing your first Serviceguard package using a monitor script!

## **9-7. LAB Solution: logprog Package Monitoring Script**

### **Directions**

1. Instead of using `/usr/lbin/logprog` as the service, create a separate service monitoring script called `/etc/cmcluster/logprog/logprog.mon` as follows:

```
# cd /etc/cmcluster/logprog

# vi logprog.mon

#!/usr/bin/sh

LOG="/etc/cmcluster/logprog/logprog.cntl.log"

echo "Now entering the logprog package monitor \c"      >> $LOG
echo "script on $(hostname) at $(date)."                  >> $LOG
while true
do
    if ps -ef | grep -v grep | grep -q "/usr/lbin/logprog"
        then
            echo "Package logprog apparently ok \c"
            echo "on $(hostname) at $(date)."
            sleep 10
        else
            echo "Package logprog failed at $(date) \c"
            echo "from node $(hostname)."
            exit
        fi
    done >> $LOG
```

The extra echo statements in the above script are sometimes referred to as "debug lines". Once the package and its scripts are working properly, these debug lines may be safely commented out. Good programming practice would suggest that these "debug lines" would be commented out (but not removed) from the script since they may be needed again in the future.

2. Make the monitor script executable.

**Answer:**

```
# chmod 755 /etc/cmcluster/logprog/logprog.mon
```

3. Copy the monitor script to other system in your cluster.

**Answer:**

```
# rcp logprog.mon <node2>:/etc/cmcluster/logprog/.
```

4. In the `logprog` package control script `/etc/cmcluster/logprog/logprog.cntl`, change the following two lines. To accomplish this, remove the `SERVICE_CMD[0]="/usr/lbin/logprog"` line and replace it with the following. Also, please remember to change the number of restarts to 0.

```
SERVICE_CMD[0]="/etc/cmcluster/logprog/logprog.mon"
SERVICE_RESTART[0]="-r 0"
```

In other words, Serviceguard will now be watching the `logprog.mon` process, and we will use zero restarts. In turn, the `logprog.mon` process will be watching the health of the `logprog` package itself.

5. In the `customer_defined_run_cmds` function of the `logprog` package control script, `/etc/cmcluster/logprog/logprog.cntl`, remove the line beginning with the colon (`:`). Then, in its place, enter the command to start `logprog`. Do not remove the `"test_return 51"` line.

```
/usr/lbin/logprog &
```

**Important:** Do not forget to put your `logprog` process in the background!

6. In the `customer_defined_halt_cmds` function, remove the line beginning with the colon (`:`), and also remove any blank lines. Do not remove the `"test_return 52"` line. Then, replace it (them) with the following few lines. This will stop our `logprog` if it is running:

```
function customer_defined_halt_cmds
{
LOG="/etc/cmcluster/logprog/logprog.cntl.log"

echo "Now entering the customer_defined_halt_cmds \c" >> $LOG
echo "on $(hostname) at $(date)." >> $LOG
if ps -ef | grep -v grep | grep -q "/usr/lbin/logprog"
    then
        echo "Found the logprog process at $(date)." >> $LOG
        echo "Killing logprog process at $(date)." >> $LOG
        kill -9 $(ps -ef | grep -v grep | grep "/usr/lbin/logprog" \
            | cut -c10-14)
    else
        echo "Note: In customer_defined_halt_cmds \c" >> $LOG
        echo "on $(hostname), and could not find the \c" >> $LOG
        echo "logprog process at $(date)." >> $LOG
    fi
```

Once again, the extra echo statements in the above code are sometimes referred to as "debug lines". Once the package and its scripts are working properly, these debug lines may be safely commented out but not removed (as discussed above in step 1).

**NOTE:** If Serviceguard believes the halt script did not properly stop the application, the halt script will fail, and Serviceguard will not start the package on the adoptive node to avoid risking application data corruption.

---

7. Copy the revised package control script to node2.

**Answer:**

```
# rcp /etc/cmcluster/logprog/logprog.cntl
<node2>:/etc/cmcluster/logprog
```

8. Verify that all the pieces work. Open a terminal window and execute tail -f /etc/cmcluster/logprog/logprog.cntl.log to view the package activities.

**Answer:**

```
# tail -f /etc/cmcluster/logprog/logprog.cntl.log
```

9. Halt the cluster and then verify that the logprog program has also exited.

**Answer:**

```
# cmhaltcl -v -f
# ps -ef | grep logprog      ( if necessary, kill the PID for the logprog process )
```

10. Now, restart the cluster, which automatically restarts the logprog package.

**Answer:**

```
# cmrunc1 -v
```

11. Kill the `logprog` process ( one time ) and verify it restarts on the adoptive node.

**Answer:**

```
# ps -ef | grep logprog
# kill <pid>

# cmviewcl -v -p logprog
```

12. Verify that the echo message was written to the package log file on the failing node.

**Answer:**

```
# vi logprog.cntl.log
```

13. Return the `logprog` package to the primary node by halting, then immediately restarting the cluster.

**Answer:**

```
# cmhaltcl -v -f
# cmrunc1 -v
```

14. Congratulations on building and testing your first Serviceguard package using a monitor script!

Module 9  
**Application Monitoring Scripts**

---

## **Module 10 — Best Practices**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- View and interpret current activities within the cluster.
- View cluster-related activities over time.
- View activities performed by the `run` and `halt` scripts.
- Use contributed Serviceguard commands to perform troubleshooting.
- Understand the Safety Time Register
- Replace a failed LVM lock disk

## 10-1. SLIDE: Best Practices

### Best Practices



- Important log file information
- Resolve cluster and package configuration issues
- Test cluster operations
- Contributed commands
- Other considerations

### Student Notes

Using "best practices" involves all the items on the above slide, including knowing which log files to monitor, knowing how to interpret and resolve configuration problems, and knowing what commands ( including the contributed commands ) to use to further diagnose problems.

This module covers the above topics.

## 10-2. SLIDE: Monitoring the syslog File

### Monitoring the syslog File



```
# tail -f /var/adm/syslog/syslog.log

May 2 15:30:51 sysA cmclld[824]: Clvmd initialized successfully
May 2 15:30:53 sysA cmclld[824]: Executing '/etc/cmcluster/pkg.cntl'
May 2 15:30:54 sysA LVM[3460]: vgchange -a e /dev/vg01
May 2 15:30:57 sysA CM-pkg[956]: cmrurunserv serv1.l>>pkg.cntl.log2>&1
May 2 15:30:57 sysA cmclld[824]: Started pkg on node sysA
May 2 15:48:44 sysA cmclld[824]: Executing '/etc/cmcluster/pkg.cntl stop'
for package pkg
May 2 15:48:49 sysA CM-pkg[824]: cmmmodnet -r -i 15.19.81.75 15.19.80.0
May 2 15:48:50 sysA LVM[3460]: vgchange -a n /dev/vg01
May 2 15:48:51 sysA cmclld[824]: Halted pkg on node sysA
May 2 15:48:52 sysA cmclld[824]: Disabled pkg on node sysA
May 2 15:48:52 sysA cmclld[824]: Package script exited with NO_RESTART
May 2 15:48:52 sysA cmclld[824]: Examine file /etc/cmcluster/pkg.cntl.log
May 2 15:48:52 sysA cmclld[824]: Switching disabled on pkg
```

### Student Notes

Use the `tail /var/adm/syslog/syslog.log` command to view events and activities that happened within the cluster in the recent past. The Serviceguard daemon `cmclld` writes status information and activities related to the cluster to the `syslog` file. Examples of entries found in the `syslog` file include the following:

- Initialization and shutdown of the cluster.
- Activation and deactivation of cluster volume groups.
- Package start and halt attempts.
- LAN card failures and recoveries.
- Cluster reformation due to kernel delays

A recommended "best practice" is to monitor the `syslog` file for occurrences of the text, "`NO RESTART`". This text string is written to the `syslog` file when a failure condition is encountered AND there is no enabled node for the package to be restarted on. This means the package will be in a DOWN state. The "`NO RESTART`" is written in all caps to indicate the urgency of this message. It is the only string which Serviceguard writes to the `syslog` file in all caps.

**Module 10**  
**Best Practices**

Kernel delays will show up as Warnings. Examples of these include the following:

```
May 2 15:50:34 sysA cmcld: Warning: cmcld process was unable to run for  
the last 8 seconds,  
May 2 15:50:34 sysA cmcld: which is longer than the node timeout  
( 5 seconds )
```

After this you normally see a cluster reformation. This indicates a kernel delay preventing cmcld from running.

### 10-3. SLIDE: Package Script Troubleshooting

Package Script Troubleshooting		
<u>Package Script Activity</u>	<u>Status</u>	
Activate Volume Group	(OK )	
File System Check and Mount	(OK )	
Add IP Address	(OK )	
Customer Defined Commands	(OK )	
Start Services	(OK )	
Activate Volume Group	(OK )	
File System Check and Mount	(OK )	
Add IP Address	(FAILED)	
-----		
Unmount File System		
Deactivate Volume Group		

### Student Notes

When a package starts, the following five major activities are performed by the package control script:

1. Activate volume group.
2. Check and mount file systems.
3. Add package IP address.
4. Execute customer-defined run commands ( start application ).
5. Start the service processes.

If a failure occurs in step 1 – 3, then the script needs to undo ( or back out of ) all previous completed steps to allow the package to restart on other nodes.

For example, if the package control script failed while adding the package IP address ( as shown in the slide ), then it is critical that the file system be unmounted and the volume group be deactivated such that the package has a chance of starting successfully on another node in the cluster. In this case, this is exactly what the package would do.

## 10-4. SLIDE: Package Script Log — Example 1

### Package Script Log — Example 1

The diagram shows a blue header bar with the title "Package Script Log — Example 1" and the HP Invent logo. Below the header is a white content area containing a terminal session output. A callout box labeled "Key on date field" has three arrows pointing to the date and time fields in the first two log entries.

```
# tail -f /etc/cmcluster/pkg/pkg.cntl.log

##### Node "sysA": Starting pkg at Sun Apr 6 21:40:33 PST
Apr 6 21:40:33 - "sysA": Activating volume group /dev/vg01 with exclus
Volume group /dev/vg01 has been successfully changed.
Apr 6 21:40:36 - "sysA": Checking filesystems:
/dev/vg01/lvol1
Apr 6 21:40:37 - "sysA": Mounting /dev/vg01/lvol1 at /mcsq_dir
/mcsq_dir: No such file or directory
ERROR: Function check_and_mount
ERROR: Failed to mount /dev/vg01/lvol1 to /mcsq_dir
Apr 6 21:40:38 - "sysA": Deactivating volume group /dev/vg01
Volume group /dev/vg01 has been successfully changed.

-----
##### Node "sysA": Starting pkg at Sun Apr 6 21:46:09 PST
Apr 6 21:46:09 - "sysA": Activating volume group /dev/vg01 with exclus
Volume group /dev/vg01 has been successfully changed.
Apr 6 21:46:13 - "sysA": Checking filesystems:
/dev/vg01/lvol1
Apr 6 21:46:13 - "sysA": Mounting /dev/vg01/lvol1 at /mcsq_dir
Apr 6 21:46:14 - "sysA": Adding IP address 15.19.81.75 to subnet
```

### Student Notes

All major activities performed by the package control script are written to the package control script log file. Each package will have its own control script log file called /etc/cmcluster/pkg/pkg.cntl.log. Each entry in the log file corresponds to one of the five activities performed by the package control script. Each entry is indexed with the date and time the specific activity was started.

### Example

The example on the slide shows a package control script that succeeds in activating the volume group and checking the file system, but fails during the mount of the file system. Through the log file, we see the script backing out of the failed attempt by deactivating the volume group.

In this example, the error occurred because the mount point directory did not exist. Once the mount point directory is created, the script is re-executed with the package successfully being started (as shown on the bottom half of the slide).

This approach of backing out of a failed package start up attempt is only used if the error occurs during steps 2 or 3. If an error occurs during steps 4 or 5, then a different approach for backing out of the failed start up attempt is used.

## 10–5. SLIDE: Package Script Log — Example 2

### Package Script Log — Example 2



```
# tail -f /etc/cmcluster/pkg/pkg.cntl.log

##### Node "sysA": Starting pkg at Sun Apr 6 21:46:09 PST
Apr 6 21:46:09 - "sysA": Activating volume group /dev/vg01 with exclus
Volume group /dev/vg01 has been successfully changed.
Apr 6 21:46:13 - "sysA": Checking filesystems:
Apr 6 21:46:13 - "sysA": Mounting /dev/vg01/lv01 at /mcsg_dir
Apr 6 21:46:14 - "sysA": Adding IP address 15.19.81.75 to subnet
SQLDBA> Connected.
SQLDBA> ORACLE instance started.
Database mounted.

ORA-01157: cannot identify data file 1 - file not found
Attempting to dismount database.....Database dismounted.
Attempting to shutdown instance.....ORACLE instance shutdown.
Apr 6 21:46:48 - "sysA": Starting service oracle_monitor
-----
##### Node "sysA": Halting pkg at Sun Apr 6 21:40:55 PST
Apr 6 21:46:55 - "sysA": Halting service oracle_monitor
ERROR: No such process with PID
SQLDBA> Connected.
SQLDBA> DBA-00342: unable to complete internal login
```

### Student Notes

The example on the slide illustrates how the script handles errors that occur during steps 4 and 5.

If the first three major steps of the package control script succeed ( volume group activation, file system checking and mounting, and adding IP address ), then the script will execute step 4 and 5 regardless of whether they are successful, and the script will no longer try to back out or undo previously successful steps.

If an error occurs during step 4 ( the start up of the customer application ) or step 5 ( the start up of the service process ), it is the responsibility of the halt script to undo all of the previous successful activities.

### Example

In the example on the slide, steps 1–3 succeed. During step 4 ( in this case, starting an Oracle application ), an error occurs. Despite the error, the script continues to step 5 and starts up the service process.

When the service process starts (in this case the Oracle monitoring script), it quickly realizes the application is not healthy and terminates. The termination causes the `cmcld` daemon to launch the halt script for the package. The halt script then backs out (undoes) all five of the major activities.

## 10–6. SLIDE: Common Cluster Configuration Issues

### Common Cluster Configuration Issues



#### Cluster-wide configuration issues:

- The `/etc/cmcluster/cmclnodelist` file does not contain names of all nodes in the cluster.
- Missing or incorrect entries in `/etc/services` or `/etc/inetd.conf`.
- Standby LAN card configured with an IP address.
- Multiple LAN cards on one node configured for the same subnet.
- LVM volume groups have non-matching minor numbers.
- Cluster lock disk not cabled to all nodes in cluster.
- An IP address is used for a nodename in `cmclconfig.ascii` file.
- Volume Group containing cluster lock disk must be activated ( or already marked for clustering ) before running the `cmaplyconf` command.

### Student Notes

The following are some common problems that may occur when forming a cluster:

- The username root must be given as the second field for each entry in `/etc/cmcluster/cmclnodelist`. The following is a sample of a properly written `/etc/cmcluster/cmclnodelist` file for a 4-node cluster:

r213c2	root
r213c3	root
r213c4	root
r213c5	root

- Entries are missing from `/etc/services` or `/etc/inetd.conf`. This is especially common when the nodes in the cluster are NIS clients. In this case, the NIS master `/etc/services` file needs updating to include the high availability service entries.
- The standby LAN card is configured. In order for Serviceguard to configure a LAN card as standby, the LAN card must not be configured with an IP address.

- If the standby LAN card has been configured with an IP address, remove all references to it in `/etc/rc.config.d/netconf` and reboot.
- The heartbeat IP is configured for the same subnet as the data IP. If a separate LAN card is used for a dedicated heartbeat, that IP address must be on a separate subnet from the primary LAN card. This is a requirement of routing protocols that become confused when two LAN cards within the same machine are on the same subnet.
- There are volume groups containing inconsistent minor numbers between two systems. The configuration of shared LVM volume groups must be the same for all systems in the cluster.
- The cluster lock is cabled to only some nodes in the cluster. The cluster lock disk must be cabled to every node in the cluster. Remember, the cluster lock is only required for two nodes, but is recommended for three or four nodes.
- An IP address (instead of a host name) is used to specify a `NODE_NAME` in cluster configuration ASCII file. All node names specified in the cluster configuration file must be host names; they cannot be the IP addresses of the nodes.
- The cluster lock disk could not be initialized. The cluster lock disk is initialized when the `cmapplyconf` command is executed. In order to initialize the cluster lock disk, the volume group must be activated. If the volume group is not activated, an attempt to activate it in exclusive mode will be made. If the attempt fails, an error is returned indicating the cluster lock disk could not be initialized. To avoid this error, either ensure the volume group is already activated prior to running the `cmapplyconf` command or make sure that the cluster lock volume group is marked for clustering.

## 10-7. SLIDE: Common Package Configuration Issues

### Common Package Configuration Issues



#### Package Configuration issues:

- Service name in `pkg.conf` does not match service name in `pkg.cntl`.
- Variable definition fields ( like `$SERVICE_CMD` ) are not quoted.
- Corrected package scripts are not propagated to other nodes in cluster.
- Mount point directory does not exist on all nodes in cluster.
- Monitor script errors ( for example, run script does not have permission to execute the monitor script ).

### Student Notes

The following are some common problems that may occur when configuring a package:

- Inconsistent service names are used in the package configuration file and package control script. The name specified in the `SERVICE_NAME` variables must be same across both the package configuration file and control script.
- String variables with spaces are not quoted. Variables within the package control script must be quoted if they contain a space as part of the string.
- Corrections to script files are not propagated to other nodes in the cluster. It is easy to forget to propagate changes to control scripts to other nodes within the cluster. Forgetting to propagate these changes will lead to errors when the package fails to other nodes.
- Mount point directories exist only on one node. Mount point directories need to exist on every node that may potentially run the package.

- There are application monitoring script errors. Application monitoring scripts contain the same issues as package control scripts: forgetting to propagate changes, string variables not being quoted, and execute permission not existing on the file.

---

## 10–8. SLIDE: Testing Cluster Operations

### Testing Cluster Operations



#### Components of the cmclld daemon which need to be tested:

- Test the Package Manager
- Test the Cluster Manager
- Test the Network Manager

### Student Notes

Once the Serviceguard system is configured, the above components should be tested to ensure that they behave correctly. The following procedures below test that the cluster responds properly in the event of a package failure, a node failure, or a LAN failure.

#### Testing the Package Manager

To test that the package manager is operating correctly, perform the following procedure for each package on the cluster.

1. Obtain a PID of a service process:

```
# ps -ef | grep <service_process>
```

2. Kill the PID of the service process:

```
# kill <PID_service_process>
```

3. View the package status:

```
# cmviewcl -v -p <package_name>
```

4. Once you have verified the package fails over successfully, move the package back to the primary node:

```
# cmmodpkg -v -e -n <primary_node> <package_name>
# cmhaltpkg -v <package_name>
# cmrungpk -v <package_name>
# cmmodpkg -v -e <package_name>
```

## Testing the Cluster Manager

To test that the cluster manager is operating correctly, perform the following procedure.

1. Power off one of the nodes in the cluster:
2. Monitor the status of the cluster; ensure that the cluster reforms:

```
# cmviewcl -v
```

Ensure that the powered off node is in a halted state, and that its packages have been restarted.

3. Restore power to the halted node.
4. Monitor the status of the cluster; ensure that the node rejoins the cluster:

```
# cmviewcl -v
```

The node should be recognized by the cluster, but the packages should not be running on the node.

5. Move the package( s ) back to the node:

```
# cmhaltpkg -v <package_name>
# cmrungpk -v <package_name>
# cmmodpkg -v -e <package_name>
```

## Testing the Network Manager

To test that the network manager is operating correctly, perform the following procedure for each node in the cluster:

1. Identify the primary and standby LAN cards on the node:

```
# lanscan
# cmviewcl -v
```

2. Disconnect the LAN from the primary LAN card.
3. View the status of the LAN cards; the local switch should have occurred so that the primary LAN card is down, and the standby LAN card is up.

```
# cmviewcl -v
```

Module 10  
**Best Practices**

4. Reconnect the LAN to the original primary LAN card. Verify the primary LAN card is back up.

```
# cmviewcl -v
```

## 10–9. SLIDE: Useful Troubleshooting Commands

### Useful Troubleshooting Commands



cmquerycl	– Verify and troubleshoot cluster and packages
cmcheckconf	– Verify and troubleshoot cluster and packages
cmscanc1	– Creates a report regarding resources in the cluster
cmviewconf	– Extracts data from the binary file, even when the cluster is not running
cmsetlog	– Set logging levels for the cmcld cluster daemon

This command is contributed; no support.

### Student Notes

The above slide shows a number of commands which can be used to collect information about the cluster.

#### The `cmquerycl` and `cmcheckconf` Commands

In addition to being able to verify cluster configuration information, the `cmquerycl` and `cmcheckconf` commands can also be used to troubleshoot the cluster.

These two commands check:

- The network addresses and network connections
- The cluster lock disk connectivity
- The validity of configuration parameters in the cluster and package configuration files

### The `cmsetlog` Command

The `cmsetlog` command is used to modify the amount of information logged by the `cmcld` cluster daemon. The amount of information logged may be set by functionality area or for certain categories of errors.

The syntax for the `cmsetlog` command is

```
cmsetlog [-M <module>] [-C <category>] [log level] [[-r] | [-s]] [-f file]
```

in which

- r                Resets logging to defaults.
- s                Stops logging to file and logs just to `syslog.log`
- f file          Specifies a log file.
- M <module>     Specifies a module in the cluster daemon to modify. If no modules are specified, all modules are selected. The following modules can be specified:
  - CLM              cluster management
  - PKG              package management
  - NET              network interface
  - SRV              service monitoring
  - LOC              local communications
  - REM              remote communications
  - SDB              status database
  - RES              resources
  - SYN              synchronization
  - CSV              command server
  - COM              communications server
  - DLM              distributed lock manager
  - GMS              ops group membership service
  - LVM              shared LVM
  - UTL              general support
  - CDB              configuration database
  - STA              status database API
  - DEV              storage devices
  - QSM              quorum devices
  - SEC              security
  - UTD              utility daemons
  - CMP              config/status proxy
- C <category>   Specifies the category of messages to modify. If no categories are specified, all categories are selected. The following categories can be specified:
  - ERR              Internal errors. Logs internal error events detected.
  - XER              External errors. Logs external error events detected.
  - DTH              Fatal errors. Logs events which cause `cmcld` to terminate.

INT	Internal events. Logs general Serviceguard events.
EXT	External events. Logs general nonServiceguard events.
PER	Periodic events. Logs heartbeats and other monitoring events.
<level>	Specifies the level of logging to use when writing to the <code>syslog.log</code> file. The lower levels provide less information, the higher levels provide more information.
0	Minimal logging (default)
1	Logs additional information on event handling
2	Logs basic logic flow (state changes) with minimal detail
3	Logs enhanced logic flow with moderate detail
4	Logs complete logic flow with maximum detail
5	Logs function entry and exit
6	Logs intrafunction execution flow (records a lot of data)

### The `cmscanc1` Command

The `cmscanc1` command displays information about all the nodes in a cluster in a structural report that allows you to compare such items as IP addresses or subnets, physical volume names for disks, and other node-specific items for all nodes in the cluster. The `cmscanc1` command actually executes several different HPUX commands on all nodes and gathers the output into a report on the node where you run the command.

The data that is displayed by `cmscanc1` follows:

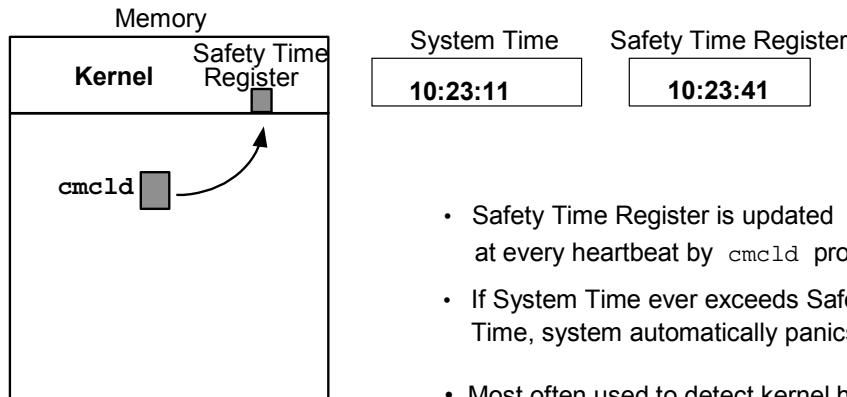
Description	Source of Data
LAN device configuration and status	<code>lanscan</code> command
network status and interfaces	<code>netstat</code> command
file systems	<code>mount</code> command
LVM configuration	<code>/etc/lvmtab</code> file
LVM physical volume group data	<code>/etc/lvmpvg</code> file
link level connectivity for all links	<code>linkloop</code> command
binary configuration file	<code>cmviewconf</code> command

### The `cmviewconf` Command

The `cmviewconf` command allows for the examination of the binary configuration file, even when the cluster is not running. The command displays the content of the binary file, `/etc/cmcluster/cmclconfig`, on the node where the command is executed.

## 10-10. SLIDE: The Built-In Safety Net

### The Built-in Safety Net



- Safety Time Register is updated at every heartbeat by `cmcld` process.
- If System Time ever exceeds Safety Time, system automatically panics.
- Most often used to detect kernel hangs

### Student Notes

Safety time is a facility provided by the kernel to ensure a specified time period is never reached. If the predefined safety time is reached, the kernel panics the system rather than continuing to process past the safety time marker.

Serviceguard uses the safety time register to ensure the `cmcld` daemon is always running. If this daemon ever terminates, it is preferable for the system to TOC rather than jeopardize the integrity of the system. Every **HEARTBEAT\_INTERVAL**, the `cmcld` daemon updates the safety time register to be at least 30 seconds ( based on the cluster reformation time ) ahead of the current system uptime. If the `cmcld` daemon terminates, the current system time will exceed the time in the safety register and the system will crash.

Serviceguard also uses the safety time register to detect kernel hangs.

## 10-11. SLIDE: Lost Cluster Lock Disk (LVM Disks)

### Lost Cluster Lock Disk ( LVM Disks )

Question: What happens when the cluster reforms?

### Student Notes

We have already discussed the need for a lock disk. This slide focuses on using LVM disks to provide cluster lock services. If you are using a Quorum Server to provide cluster lock services, then the same discussion applies, except that we would be discussing the loss of the Quorum Server.

If the cluster splits exactly in half the lock disk is used as a tie-breaker to determine which half continues to run the packages and which half TOCs. If the cluster splits in half and the lock disk is not available then both halves will TOC and all packages will be down.

If the lock information is housed on a RAID 1 or RAID 5 LUN in a disk array, the loss of a single disk does not cause the lock information to be unavailable because the lock information is protected. Simply replace the broken disk and the array will automatically rebuild the data as usual.

Note: The user data on the lock disk was still available via a mirror copy of the data on another disk. On LVM mirrored disks as we are using in the classroom, the cluster lock resides in the BBRA of the lock disk. This is an LVM reserved area that is not mirrored.

Module 10  
**Best Practices**

If the lock information is housed on a LVM disk ( and not in a RAID 1 or RAID 5 LUN ) and the lock disk fails we have extra manual steps that must be taken to reinitialize the lock information.

### **Steps to Recover a Lost Cluster Lock Disk**

- Step 1: Replace the defective disk.
- Step 2: Execute `vgcfgrestore` to rebuild the LVM reserved areas on this disk.  
Note: even though the lock information is stored in the LVM reserved area, the `vgcfgrestore` command does not reinitialize the lock information. The lock information is stored in the bad block relocation table. There wouldn't be any reason for `vgcfgrestore` to replace the BBRA, since it's likely we replaced the disk anyhow.
- Step 3: If the disk also has logical volumes, follow the standard procedures to either rebuild them or sync them with the mirror.
- Step 4: Serviceguard checks the lock disk on an hourly basis. After the `vgcfgrestore` command, review the syslog file of an active cluster node for not more than one hour and look for a message that the lock disk is healthy again.

---

**NOTE:** If the `vgcfgbackup` command was not done on the lock disk volume group after the cluster was formed, the following steps must take place to restore the cluster lock:

---

1. Stop the cluster.
2. Execute the `capplyconf` command on the cluster config file.
3. Restart the cluster.
4. Back up the volume group that contains the cluster lock using `vgcfgbackup`.

## 10-12. SLIDE: Monitoring Cluster and Package Resources

### Monitoring Cluster and Package Resources

Question: Will the package detect when the mirror fails?

### Student Notes

In general, in a high availability environment, virtually any single component can fail, and all applications should still be accessible when they do. In some cases, applications may need to restart on their adoptive nodes, in other cases they may continue on the primary.

For example, if a single disk fails, the data housed on that disk should still be available either on the LVM mirror copy or on the RAID 1 or RAID 5 LUN.

Not only is it necessary to build redundancy into the environment, but it is also necessary to know when a failure has occurred. If we do not detect a failure, and continue on the redundant device (eg. a mirror disk or standby LAN card), we expose ourselves to dual-points of failure. We must also constantly monitor for any kind of failure.

If a primary disk failure goes undetected by administrators, eventually the mirror copy will also fail, and the users will experience extended unplanned downtime. Presumably this is unacceptable, so all failures of any kind must be constantly monitored and corrected before the redundant component also fails.

**Module 10**  
**Best Practices**

Items that should be monitored include:

- Disk
- Node
- Package
- LAN controllers
- Routers ( and other network components )
- UPS
- Free disk space in file systems
- Performance
- Any component or resource the users need

To monitor some or all of the above involves one or more of the following:

- writing monitoring scripts,
- implementing OpenView Network Node Manager,
- implementing ITO,
- using SGMgr
- implementing EMS.

## 10-13. SLIDE: Notification for Package Failure

### Notification for Package Failure

**Objective:**

**When package halts on primary or starts on adoptive node, notify the operations staff.**

START

```
if [ $(hostname) = "adoptive" ]
then
    notify admin ( page, email, etc )
fi
```

HALT

```
if [ $(hostname) = "primary" ]
then
    notify admin ( page, email, etc )
fi
```

### Student Notes

In a highly available environment it is essential that administrators are quickly notified of any failure or abnormality. A recommended practice would be to build admin notification into the start/halt scripts. If the package halts on the primary or starts on the adoptive node the administrator should be notified. The script can test for which node it's on and then page or e-mail the administrator.

#### Special Note

In the above slide, it is understood that we would not type in the word "adoptive", or the word "primary". Rather, we would type in the actual hostname of the machine which is the adoptive node or the actual hostname of the machine which is the primary node.

## **10-14. LAB: Package Monitoring with Email Notification**

### **Directions**

1. Add the code to the `logprog` package `run / halt` script so that when the package stops on the primary it mails the message:

```
"logprog package has stopped on primary node"
```

to root on the primary system. For maintenance purposes, it is recommended that the `run / halt` script be identical on both systems.

2. Add the code to the `logprog` package `run / halt` script so that when the package starts on the adoptive node it mails the message:

```
"logprog package has started on adoptive node"
```

to root on the adoptive system. Again, for maintenance purposes, it is recommended that the `run / halt` script be identical on both systems.

3. Copy the `logprog.cntl` and `error.msg1` and `error.msg2` files to `node2`.

4. Failover the `logprog` package and verify the mail messages are properly sent.

5. Return the `logprog` package to the primary node.

## 10-15. LAB: Using the `cmsetlog` Command

### Directions

Complete the following exercises.

1. Set the logging level of the `cmcld` daemon to a 1 for the cluster manager module. Only use a level of 1 for periodic events generated by the cluster manager. Answer "Y" to the questions.
2. Continuously tail the end of the `/var/adm/syslog/syslog.log` file and note if any additional information is being added.
3. Reset the logging level to its default settings (so that the `syslog` file does not get so big that it fills up the file system).

## 10-16. LAB Solution: Package Monitoring with Email Notification

### Directions

1. Add the code to the `logprog` package `run / halt` script so that when the package stops on the primary it mails the message:

```
"logprog package has stopped on primary node"
```

to root on the primary system. For maintenance purposes, it is recommended that the `run / halt` script be identical on both systems.

#### Answer:

- a. In the `customer_defined_halt_cmds` function, add the following code:

```
if [ $(hostname) = "primary_node_name" ]
then
    mail root < /etc/cmcluster/logprog/error.msg1
fi
```

- b. Create the file `error.msg1` and replace the "`primary_node_name`" above with your primary node name.

```
# vi error.msg1
```

```
logprog package has stopped on primary node
```

2. Add the code to the `logprog` package `run / halt` script so that when the package starts on the adoptive node it mails the message:

```
"logprog package has started on adoptive node"
```

to root on the adoptive system. Again, for maintenance purposes, it is recommended that the `run / halt` script be identical on both systems.

#### Answer:

- a. In the `customer_defined_run_cmds` function, add the following code:

```
if [ $(hostname) = "adoptive_node_name" ]
then
    mail root < /etc/cmcluster/logprog/error.msg2
fi
```

- b. Similar to the above, an `error.msg2` file needs to be created. Then replace the "`adoptive_node_name`" above with the name of your adoptive system.

```
# vi error.msg2
```

```
logprog package has started on adoptive node
```

3. Copy the `logprog.cntl` and `error.msg1` and `error.msg2` files to node2.

**Answer:**

```
# rcp logprog.cntl <node2>:/etc/cmcluster/logprog/.  
# rcp error.msg1 <node2>:/etc/cmcluster/logprog/.  
# rcp error.msg2 <node2>:/etc/cmcluster/logprog/.
```

4. Failover the `logprog` package and verify the mail messages are properly sent.

**Answer:**

```
# ps -ef | grep logprog  
# kill <pid>  
# elm
```

5. Return the `logprog` package to the primary node.

**Answer:**

```
# cmhaltpkg -v logprog  
# cmmodpkg -v -e -n <primary_node> logprog  
# cmmodpkg -v -e -n <adoptive_node> logprog  
# cmmodpkg -v -e logprog
```

## 10-17. LAB Solution: Using the cmsetlog command

### Directions

Complete the following exercises.

1. Set the logging level of the `cmcld` daemon to a 1 for the cluster manager module. Only use a level of 1 for periodic events generated by the cluster manager. Answer "y" to the questions.

**Answer:**

```
# cmsetlog -M CLM -C PER 1
```

2. Continuously tail the end of the `/var/adm/syslog/syslog.log` file and note if any additional information is being added.

**Answer:**

```
# tail -f /var/adm/syslog/syslog.log
```

3. Reset the logging level to its default settings (so that the `syslog` file does not get so big that it fills up the file system).

**Answer:**

```
# cmsetlog -r
```

---

# **Module 11 — Cluster and Package Online Reconfiguration**

## **Objectives**

Upon completion of this module, you will be able to do the following:

- Change the cluster configuration while the cluster is running.
- Add a node to the cluster while the cluster is running.
- Remove a node from the cluster while the cluster is running.
- Add a package to a running cluster.
- Delete a package from a running cluster.
- Modify specific package attributes while the package is running.
- Modify a down package while the cluster and other packages are running.
- Perform the above operations by using the HP-UX command line.

## 11-1. SLIDE: Serviceguard Online Reconfiguration

### Serviceguard Online Reconfiguration



- Ability to change cluster configuration while the cluster is running
- Online Node Functionality
  - Add a new node.
  - Remove an existing node.
- Online Package Functionality
  - Add a package.
  - Delete a package.
  - Modify some package attributes while package is running.
  - Modify entire package while cluster and other packages continue to run, but modified package is down.
  - Change Package Access Control Policy

### Student Notes

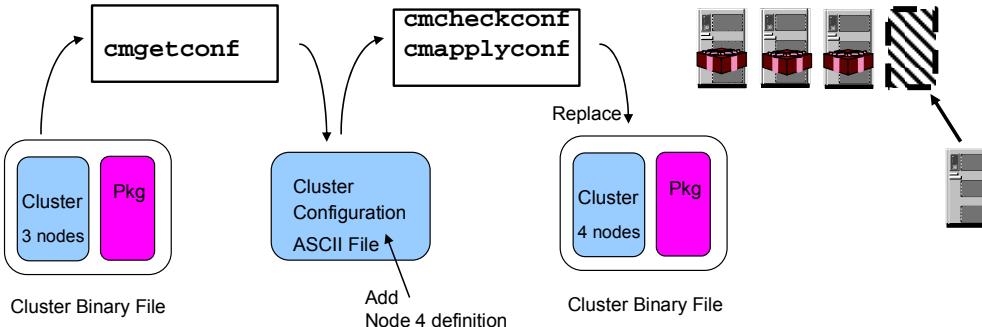
This module covers cluster and package maintenance tasks that can be performed while the cluster is up and running. Some of the tasks are related directly to cluster modifications, and other tasks focus more on package modifications. In some cases, a package can actually be reconfigured while the package itself is running. With "On-Line Modifications", only certain changes are permitted, and these modifications will be highlighted in the pages that follow.

## 11-2. SLIDE: Add a Node while a Cluster Is Running

### Add a Node while a Cluster Is Running



- Update cmclnodelist on all nodes in the cluster (see notes below)
- cmgetconf -v -c cluster1 temp.ascii
- cmquerycl -v -n n1 -n n2 -n n3 -n n4 -C cmclconfig.ascii
- Combine the two configurations into one file. ( cmclconfig.ascii )
- cmcheckconf -v -C cmclconfig.ascii
- cmapplyconf -v -C cmclconfig.ascii
- cmrunnode -v n4
- Modify package files to reflect new node
- If using a quorum server, update the qs\_authfile



```

graph LR
    subgraph CB1 [Cluster Binary File]
        direction LR
        C1[Cluster] --- P1[Pkg]
    end
    subgraph CC [Cluster Configuration ASCII File]
        direction TB
        C2[cmgetconf] --> CC
        C3[cmquerycl] --> CC
        CC --> C4[cmcheckconf  
cmapplyconf]
        C4 -- Replace --> CB2[Cluster Binary File]
        C4 -- "Add Node 4 definition" --> CB2
    end
    subgraph CB2 [Cluster Binary File]
        direction LR
        C5[Cluster] --- P2[Pkg]
        C5 --- N4[Node 4 definition]
    end

```

### Student Notes

Remember to install Serviceguard on the new node before adding it to the cluster. Also, if the cluster is running a version of Serviceguard pre-A.11.16, create the /etc/cmcluster/cmclnodelist file on the new node, populating it with all nodes in the cluster. Be sure to add the new node to all of the existing nodes' /etc/cmcluster/cmclnodelist files, as well. If the cluster is running Serviceguard A.11.16 or later, this step should be skipped. Be sure to perform the above steps before doing the cmquerycl to add the new node to the cluster.

The above example demonstrates the Serviceguard commands and the order of such commands that are used to add a new node to an existing cluster. For the above example, nodes n1, n2, and n3 are already configured in a running cluster named mycluster, and the desired action is to add node n4. This is all performed without interrupting a single event that is occurring on the existing cluster.

The first step probes the existing cluster and builds an ASCII file, temp.ascii, containing the configuration information for the existing cluster. The next step is used to generate a new template file, cmclconfig.ascii, for the desired cluster configuration. Remember to specify ALL nodes on the command line. The third step may be the most difficult. Compare cmclconfig.ascii with temp.ascii and make any necessary changes to

## **Cluster and Package Online Reconfiguration**

`cmclconfig.ascii` so the desired configuration is captured. The next step is to verify the new configuration. The fifth step applies the changes to the configuration and sends the new binary configuration file to all cluster nodes. Next, start the new node. Remember to change any/all parameter values in the necessary start-up files.

If using a quorum server, be sure to update the quorum server  
`/etc/cmcluster/qs_authfile` to reflect the new node in the cluster.

Also, be sure to run `/usr/lbin/qs -update` to force the quorum server to re-read the updated `qs_authfile`.

### 11-3. SLIDE: Remove a Node while a Cluster Is Running

## Remove a Node while a Cluster Is Running

hp invent

- cmhalt pkg -v pkg\_name
- Remove any references to node 4 from all pkg configuration files
- cmhaltnode -v node4
- Edit the cluster configuration file and remove node 4's definition.
- cmcheckconf -v -C cmclconfig.ascii
- cmapplyconf -v -C cmclconfig.ascii
- cmcheckconf -v any altered package configuration files
- cmapplyconf -v any altered package configuration files

The diagram illustrates the process of removing a node from a cluster while it is running. It shows a cluster binary file containing a cluster and packages, and a cluster-wide configuration ASCII file. The configuration file is edited to remove the definition of node 4. This leads to a new cluster binary file where the cluster now has 3 nodes instead of 4. A physical cluster icon shows four nodes, with the fourth node being crossed out, indicating it has been removed.

### Student Notes

The above example demonstrates the Serviceguard commands and the order of such commands that are used to delete a node from an existing cluster. In the example, nodes n1, n2, n3, and n4 are all members of a running cluster. The desired action is to remove node n4 from the cluster.

The first step is to remove any package reference to node n4. Therefore, packages may need to be reconfigured. Next, halt the node to be removed. Edit the cluster ASCII file and remove all configuration information referring to node n4. Now verify the new configuration and apply the changes, which will also distribute the new binary configuration file to the remaining nodes.

If you are attempting to remove an unreachable node that has many packages dependent on it, especially if the dependent packages use a large number of EMS resources, you may see the following message:

The configuration change is too large to process while the cluster is running. Split the configuration change into multiple requests or halt the cluster.

In this situation, you must halt the cluster to remove the unreachable node.

## 11-4. SLIDE: Summary of Cluster Configuration Changes

### Summary of Cluster Configuration Changes



As of SG version 11.17, the following summarizes available cluster changes:

Change to Cluster Configuration	Required Cluster State
Add a new node	All cluster nodes must be running.
Delete an existing node	A node can be deleted even if it is unavailable or unreachable.
Change Maximum Configured Packages	Cluster may be running
Change Timing Parameters	<i>Cluster must not be running.</i>
Change Cluster Lock or Quorum Server Configuration	<i>Cluster must not be running.</i>
Change serial device files	<i>Cluster must not be running.</i>
Change IP addresses for heartbeats or monitored subnets	<i>Cluster must not be running.</i>
Access control modifications	Cluster may be running

### Student Notes

See the “Managing Serviceguard” document on <http://docs.hp.com> for the complete discussion of possible cluster configuration changes.

New nodes can be added to the cluster configuration or existing nodes can be deleted from the cluster configuration while the cluster is up and running. Note the following, however:

- You cannot change the lock disk configuration while the cluster is running.
- You cannot remove an active node from the cluster. You must halt the node first.
- You cannot delete an active volume group from the cluster configuration. You must halt any package that uses the volume group and ensure that the volume group is inactive before deleting it.
- You cannot change cluster timing parameters with the cluster running.

- The only configuration change allowed while a node is unreachable (for example, completely disconnected from the network) is to delete the unreachable node from the cluster configuration.

If there are also packages that depend upon that node, the package configuration must also be modified to delete that node.

Changes to the package configuration are covered in the next slide.

## 11-5. SLIDE: Add a Package while a Cluster Is Running

### Add a Package while a Cluster Is Running

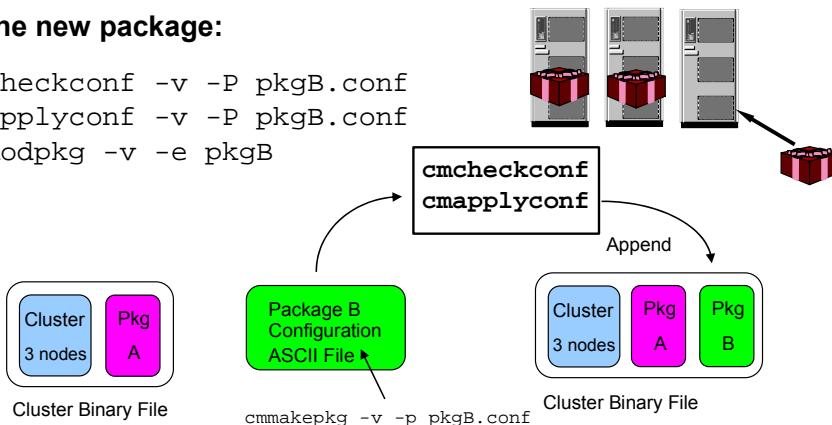


#### Normal package creation:

```
cmmakepkg -v -p pkgB.conf and edit as usual  
cmmakepkg -v -s pkgB.cntl and edit as usual  
chmod +x pkgB.cntl  
rcp pkgB.cntl node2:/etc/cmcluster/pkgB/pkgB.cntl
```

#### Apply the new package:

```
cmcheckconf -v -P pkgB.conf  
cmapplyconf -v -P pkgB.conf  
cmmodpkg -v -e pkgB
```



### Student Notes

It is possible to create a new package and add it to the cluster configuration while the cluster is up and while other packages are running. The number of packages you can add is subject to the value of `MAX_CONFIGURED_PACKAGES` as defined in the cluster-wide ASCII file.

To create the package, follow the steps for creating a package covered in the previous module with the following difference: do not specify the cluster name when verifying and distributing the configuration.

Remember to copy the control script to the `/etc/cmcluster/pkg_name` directory on all nodes that can run the package.

Adding a package to a running cluster is useful as your systems grow. Rather than purchasing a new set of systems, take advantage of the existing cluster configuration. This is especially useful when going from a 2-node active/standby configuration with an idle-node to a 2-node active/active configuration.

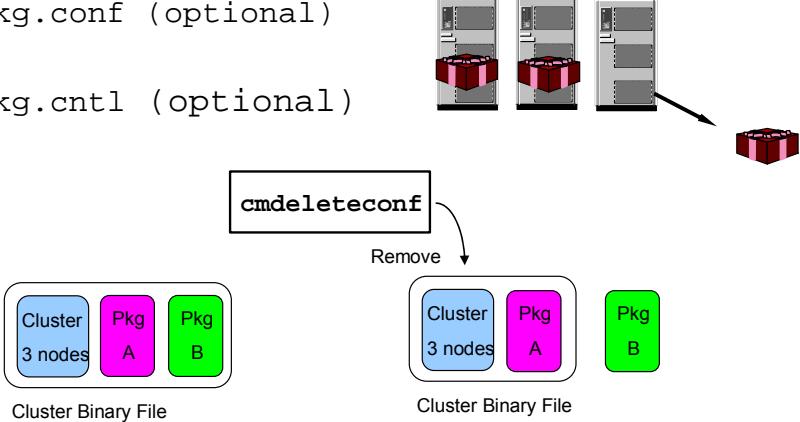
Note that any new hardware required for the newly added package must already be on all the systems for that package. For example, if adding a new package which requires a new FDDI network interface, the entire cluster would need to be brought down to install the FDDI network interface on each node in the cluster.

## 11-6. SLIDE: Remove a Package while a Cluster Is Running

### Remove a Package while a Cluster Is Running



- cmhaltpkg -v pkgB
- cmdeleteconf -v -p pkgB
- rm pkg.conf (optional)
- rm pkg.cntl (optional)



### Student Notes

A package can be deleted from all cluster nodes by using the `cmdeleteconf` command. `cmdeleteconf` can only be executed when the package is not running; however, the cluster may be up. The command removes the package information from the binary configuration file on all the nodes in the cluster. The package configuration and control files are not removed.

Deleting a package is useful when you want to make major changes to a package. Rather than modifying the package, it might be easier to simply start with a new package by deleting the old one and adding the new one.

**Caution:** `cmdeleteconf`, without options, prompts to remove the entire cluster.

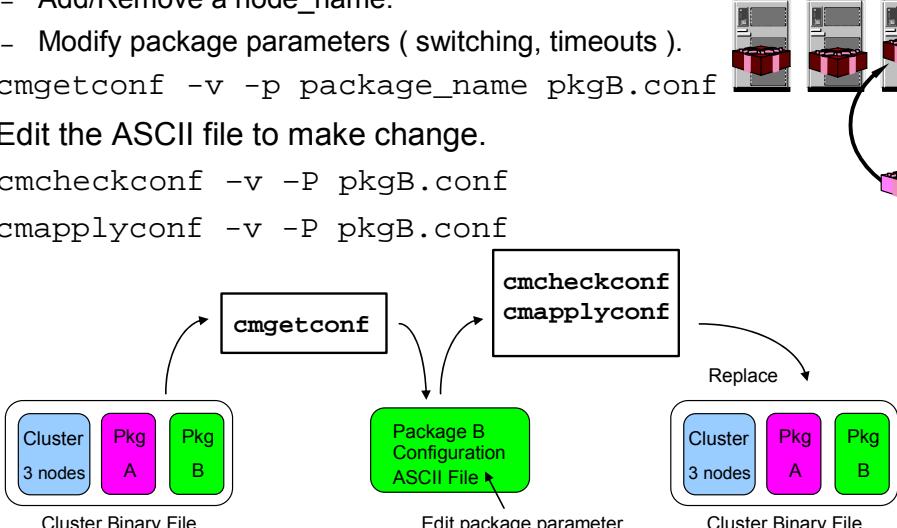
Syntax: # `cmdeleteconf -p pkg_name`

## 11-7. SLIDE: Modify a Package while the Cluster and Package Are Running

### Modify a Package while the Cluster and Package Are Running



- What can be modified without bringing down the package?
  - Modify node failover order.
  - Add/Remove a node\_name.
  - Modify package parameters ( switching, timeouts ).
- cmgetconf -v -p package\_name pkgB.conf
- Edit the ASCII file to make change.
- cmcheckconf -v -P pkgB.conf
- cmapplyconf -v -P pkgB.conf



The diagram illustrates the workflow for modifying a package while the cluster is running. It shows a 'Cluster Binary File' containing 'Cluster 3 nodes', 'Pkg A', and 'Pkg B'. The 'cmgetconf' command is used to extract configuration data from the binary file. The 'cmcheckconf/cmapplyconf' command is used to validate and apply changes to the configuration. An 'Edit package parameter' step involves modifying the 'Package B Configuration ASCII File'. Finally, the 'Replace' step updates the 'Cluster Binary File' with the modified configuration.

### Student Notes

The items on the above slide can be performed on a package while the package is running. This implies that the cluster is also running. All nodes in the cluster must be powered up and accessible when making configuration changes.

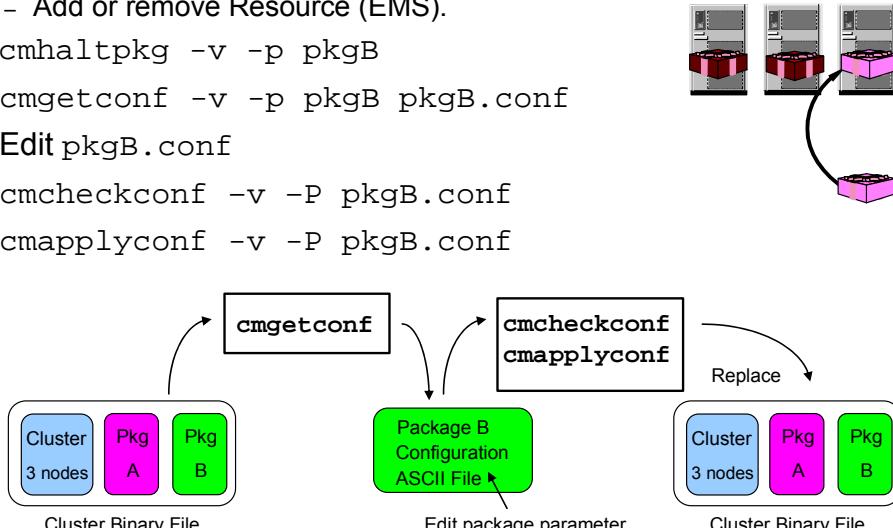
## 11-8. SLIDE: Modify a Package while the Cluster Is Running, but the Package Is Down

### Modify a Package while the Cluster Is Running, but the Package Is Down



When do I need to bring the package down?

- Add or remove Service.
- Add or remove Subnet.
- Add or remove Resource (EMS).
- `cmhaltPKG -v -p pkgB`
- `cmgetconf -v -p pkgB pkgB.conf`
- **Edit** `pkgB.conf`
- `cmcheckconf -v -P pkgB.conf`
- `cmaplyconf -v -P pkgB.conf`



### Student Notes

There are some instances when the package must be brought down. Fortunately, the cluster and remaining packages can remain up. To add a subnet or a resource to the package, the subnet or service must already be configured in the cluster. The next page provides a summary of the changes and the necessary state of a package and / or the cluster.

## 11–9. SLIDE: Summary: Package Modifications

### Summary: Package Modifications



As of SG version 11.17, the following summarizes available package changes:

Package Modification	Required Package State
Add a new package	Other packages may be UP
Add a new node	Package( s ) may be UP
Remove a node	Package( s ) may be running on different nodes
Add / Remove a service	Package must be <i>halted</i>
Add / Remove a subnet	Package must be <i>halted</i>
Add / Remove a resource	Package must be <i>halted</i>
Change run / halt script contents	Recommended that package be <i>halted</i> . Timing problems may occur if the script is modified while the package is running
Script timeouts	Package may be UP
Service timeouts	Package must be <i>halted</i>
Failfast parameters	Package must be <i>halted</i>
AutoRun	Package may be UP
Local LAN Failover	Package may be UP
Change node adoption order	Package may be UP
Access control modifications	Package may be UP
Failover / Fallback Policy	Package may be UP

### Student Notes

All nodes in the cluster must be powered up and accessible when making configuration changes. The above table summarizes package maintenance task and package state requirements.

See the “Managing Serviceguard” document for the complete discussion of possible package configuration changes.

This document is available at no cost from <http://docs.hp.com>

## 11-10. LAB: Online Package Addition

### Directions

#### Special Notes

- a. In this lab, we create an additional package, called "test", without bringing down the cluster.
- b. This new package will also be our first package to use disk resources. Specifically, we will use the vg01 volume group that was previously created in Module 4.
- c. The new package should execute the "test\_program" script which will write to the shared volume group vg01.

### Part 1: Create the Package Configuration File

1. Make a directory for the package files.

**Answer:**

```
# cd /etc/cmcluster
# mkdir test
```

2. Change directory to the package directory that was just created.

**Answer:**

```
# cd test
```

3. Create the package configuration file template.

**Answer:**

```
# cmmakepkg -v -p test.conf
```

Module 11  
**Cluster and Package Online Reconfiguration**

4. Edit the package configuration file template.

```
# vi test.conf
```

and modify the following parameters:

```
PACKAGE_NAME          test
NODE_NAME             <node1>
NODE_NAME             <node2>
RUN_SCRIPT            /etc/cmcluster/test/test.cntl
RUN_SCRIPT_TIMEOUT    NO_TIMEOUT
HALT_SCRIPT           /etc/cmcluster/test/test.cntl
HALT_SCRIPT_TIMEOUT   NO_TIMEOUT
SERVICE_NAME          test_service
SUBNET                x.x.x.x      ← Use netstat -in to obtain 10-net subnet
```

5. Save the changes, and exit the file.

## Part 2: Create and Distribute the Package Control Script

1. Create the package control script template.

**Answer:**

```
# cmmakepkg -v -s test.cntl
```

2. Edit the package control script template.

```
# vi test.cntl
```

First, find the volume groups section, which now reads:

```
#VG[ 0 ]= "
```

Modify this value so that it now reads:

```
VG[ 0 ]= "vg01"
```

← Don't forget to uncomment this line

Next, go to the section for logical volumes, which now reads:

```
#LV[ 0 ]= ""; FS[ 0 ]= ""; FS_MOUNT_OPT[ 0 ]= ""; FS_UNMOUNT_OPT[ 0 ]= ""; FS_FSCK_OPT[ 0 ]= ""  
#FS_TYPE[ 0 ]= "
```

Break these lines up so that each variable is on a separate line (uncomment the lines, remove the semi-colons and the blank space, then make the lines look like the following):

```
LV[ 0 ]= "  
FS[ 0 ]= "  
FS_MOUNT_OPT[ 0 ]= "  
FS_UNMOUNT_OPT[ 0 ]= "  
FS_FSCK_OPT[ 0 ]= "  
FS_TYPE[ 0 ]= "
```

Then, modify the following parameters:

```
LV[ 0 ]= "/dev/vg01/sglvol1"  
FS[ 0 ]= "/sgdata1"  
FS_MOUNT_OPT[ 0 ]= ""  
FS_UNMOUNT_OPT[ 0 ]= ""  
FS_FSCK_OPT[ 0 ]= ""  
FS_TYPE[ 0 ]= "vxfs"
```

## Module 11

### Cluster and Package Online Reconfiguration

Next, go to the “Service Names and Commands” section, which now reads:

```
#SERVICE_NAME[ 0 ]= " "
#SERVICE_CMD[ 0 ]= " "
#SERVICE_RESTART[ 0 ]= " "
```

Modify this section to so that it now reads ...

```
SERVICE_NAME[ 0 ]= "test_service"           ← Do not forget
SERVICE_CMD[ 0 ]= "/etc/cmcluster/test/test.mon" ← to uncomment
SERVICE_RESTART[ 0 ]= "-r 0"                  ← these lines.
```

3. In the `customer_defined_run_cmds` function in the control script, replace the colon (`:`) with the following code. We use this code to start the `test_program` script:

```
/etc/cmcluster/test/test_program &
```

Also, in the `customer_defined_halt_cmds` function, replace the colon (`:`) with:

```
/etc/cmcluster/test/kill_test
```

where `kill_test` will contain the script for terminating the `test` package. We will write this script in Part 3 of this lab.

4. Save the changes, exit the file, and make it executable.

**Answer:**

```
# chmod +x test.cntl
```

5. Now, copy the `test.cntl` control script to other nodes within the cluster.

**Answer:**

```
# remsh <node2> "/usr/bin/mkdir /etc/cmcluster/test"
# scp test.cntl <node2>:/etc/cmcluster/test/.
```

## Part 3: Create and Distribute the Package Monitor and Kill Scripts

1. Create the test script monitor `/etc/cmcluster/test/test.mon`, as follows:

```
# vi /etc/cmcluster/test/test.mon
```

```
#!/usr/bin/sh

LOG="/etc/cmcluster/test/test.cntl.log"

echo Now entering the monitor on $(hostname) at $(date). >> $LOG
while true
do
    echo "Checking for the existence of the \"test_program\" script. "
    if ps -ef | grep -v grep | grep -q "/etc/cmcluster/test/test_program"
    then
        echo Found \"test_program\" on $(hostname) at $(date).
        sleep 30
    else
        echo "Could NOT find \"test_program\" on $(hostname) at $(date). "
        echo "The monitor is now exiting on $(hostname) at $(date). "
        exit
    fi
done >> $LOG
echo "We should never see this line of code. " >> $LOG
```

2. Create the script `/etc/cmcluster/test/kill_test`, as follows. This script will, as needed, kill the `test_program` script.

```
# vi /etc/cmcluster/test/kill_test
```

```
#!/usr/bin/sh

LOG="/etc/cmcluster/test/test.cntl.log"

echo Now entering \"kill_test\" on $(hostname) at $(date). >> $LOG
echo "Now searching for all occurrences of \c" >> $LOG
echo "the \"test_program\" script on $(hostname) at $(date)." >> $LOG
if ps -ef | grep -v grep | grep -q test_program
then
    for pid in $(ps -ef | grep -v grep | grep \
                "test_program" | cut -c10-14)
    do
        echo "Now trying to kill all occurrences of \c" >> $LOG
        echo "the \"test_program\" script. " >> $LOG
        if ps -ef | cut -c10-14 | grep -q "$pid"
        then
            kill -9 $pid
        fi
    done
fi
```

Module 11  
**Cluster and Package Online Reconfiguration**

```
echo "End of the \"kill_test\" program. Now exiting ..." >> $LOG
```

3. Make the test.mon and kill\_test scripts executable.

**Answer:**

```
# cd /etc/cmcluster/test
# chmod 755 test.mon
# chmod 755 kill_test
```

4. Copy both the test.mon and kill\_test scripts to the other system.

**Answer:**

```
# scp test.mon <node2>:/etc/cmcluster/test/.
# scp kill_test <node2>:/etc/cmcluster/test/.
```

## **Part 4: Create and Distribute the Test Program**

1. Create the test\_program script (in the /etc/cmcluster/test directory).

```
#!/usr/bin/sh

count=0
while true
do
    echo $(hostname) $(date) $count >> /sgdata1/test_data
    let count=count+1
    sleep 5
done
```

2. Make the test\_program script executable, and copy it to the test package directory on the other node.

**Answer:**

```
# chmod 755 test_program
# scp test_program <node2>:/etc/cmcluster/test/.
```

## Part 5: Check the Package Configuration File, Distribute the Binary File and Start the Package

1. Check the configuration file for errors.

**Answer:**

```
# cd /etc/cmcluster/test  
# cmcheckconf -v -P test.conf
```

2. Fix any errors. Once the configuration is OK, apply the new configuration.

**Answer:**

```
# cmapplyconf -v -P test.conf
```

3. Once the new configuration is applied and distributed, enable the package.

**Answer:**

```
# cmmodpkg -v -e test
```

4. Verify the /sgdata1/test\_data file is growing by doing a tail -f /sgdata1/test\_data on the node which has the file system mounted.

**Answer:**

```
# tail -f /sgdata1/test_data
```

5. Execute /etc/cmcluster/test/kill\_test to verify that the test package fails over correctly and mounts on the adoptive node. Execute tail -f /sgdata1/test\_data on the adoptive node.

**Answer:**

Node 1:

```
# /etc/cmcluster/test/kill_test
```

Node 2:

```
# tail -f /sgdata1/test_data
```

## 11-11. LAB: Testing Package FAILOVER\_POLICY

### Directions

In this lab you will change the FAILOVER\_POLICY of one of your packages from CONFIGURED\_NODE to MIN\_PACKAGE\_NODE.

#### On the Configuration Node

1. While the following package policy changes would not require halting the cluster, the following exercises are better demonstrated if we do so. Therefore, halt the cluster now.

#### Answer:

```
# cmhaltcl -v -f
```

2. Before continuing, make a backup copy of the logprog.conf and the test.conf files.

#### Answer:

```
# cd /etc/cmcluster/logprog
# cp logprog.conf logprog.conf.save
# cd ../test
# cp test.conf test.conf.save
```

3. Examine the package configuration file for both the logprog and test packages. Notice the FAILOVER\_POLICY is currently set to CONFIGURED\_NODE. This policy means that the package will be run on the first node listed in the node list, unless that node is unavailable or the package is disabled on that node.

Configure both packages so that they have the same primary node. Use Node1 as the primary node (they may already be set up this way). If you modified the configuration, check and apply the modified files.

#### Answer:

```
# cd /etc/cmcluster/logprog
# vi logprog.conf

# cd ../test
# vi test.conf
```

4. Start the cluster. Verify that both packages are running on the same node (Node1).

## Answer:

```
# cmrunc1 -v
```

- ## 5. Now, halt the cluster.

## Answer:

```
# cmhaltcl -v -f
```

6. Modify the package configuration file for `test`. Change the `FAILOVER_POLICY` to `MIN_PACKAGE_NODE`. Check and apply the modified configuration.

## Answer:

```
# cd /etc/cmcluster/test  
# vi test.conf  
# cmcheckconf -v -P test.conf  
# cmapplyconf -v -P test.conf
```

- Start the cluster. This time `test` should have started on the other node (Node2), since it had fewer packages running.

## Answer:

```
# cmrunc1 -v
```

8. Restore the backup copy of the `test` configuration file. Reapply the restored configuration.

### **Answer:**

```
# cd /etc/cmcluster/logprog
# cmhaltpkg -v logprog
# cmhaltpkg -v test
# cp logprog.conf.save logprog.conf
# cd .. /test
# cp test.conf.save test.conf
# cmcheckconf -v -P test.conf -P ../logprog/logprog.conf
# cmapplyconf -v -P test.conf -P ../logprog/logprog.conf
# cmmmodpkg -v -e logprog
# cmmmodpkg -v -e test
```

## 11-12. LAB: Testing the Cluster-Wide ASCII File

### Directions

1. Make a backup of the cluster ASCII file.

**Answer:**

```
# cd /etc/cmcluster
# cp cmclconfig.ascii cmclconfig.ascii.save
```

2. Modify the cluster ASCII file as follows:

```
# vi cmclconfig.ascii
```

Do step 2a only if you are using LVM disks for cluster lock functionality; otherwise, skip this step and go to step 2b.

- a. Change the FIRST\_CLUSTER\_LOCK\_VG to /dev/vglock (a non-existent volume group).
  - b. Modify the HEARTBEAT\_INTERVAL to 15 seconds.
  - c. Modify MAX\_CONFIGURED\_PACKAGES to 200.
  - d. Make any other changes you would like to test.
- 
3. With the cluster running, try to execute cmcheckconf on the cluster ASCII cluster configuration file. What happens?

**Answer:**

```
# cd /etc/cmcluster
# cmcheckconf -v -C cmclconfig.ascii
```

cmcheckconf reports "First cluster lock volume group /dev/vglock cannot be found in the cluster."

4. Fix the erroneous cluster lock volume group in the cluster ascii file and attempt another cmcheckconf. What happens?

**Answer:**

```
# cmcheckconf -v -C cmclconfig.ascii

cmcheckconf reports "Modifying HEARTBEAT_INTERVAL value from 3000000 to
15000000 while cluster cluster_name is running is not supported.
cmcheckconf: Unable to verify cluster file: cmclconfig.ascii.
```

5. Halt the cluster. Reissue cmcheckconf. Will Serviceguard allow these changes to be made?

**Answer:**

```
# cd /etc/cmcluster
# cmcheckconf -v -C cmclconfig.ascii

No. cmcheckconf reports "Node timeout (6.00 seconds) should be at least
two times the heartbeat interval (15.00 seconds).
Maximum configured packages must be greater than or equal to 0, and
less than or equal to 150.
cmcheckconf: Unable to verify cluster file: cmclconfig.ascii.
```

6. Restore the original copy of the cluster ASCII file and re-apply this configuration.

**Answer:**

```
# cd /etc/cmcluster
# mv cmclconfig.ascii.save cmclconfig.ascii
# cmcheckconf -v -C cmclconfig.ascii
# cmapplyconf -v -C cmclconfig.ascii
```

7. Restart the cluster.

**Answer:**

```
# cmrunc1 -v
```

Module 11  
**Cluster and Package Online Reconfiguration**

---

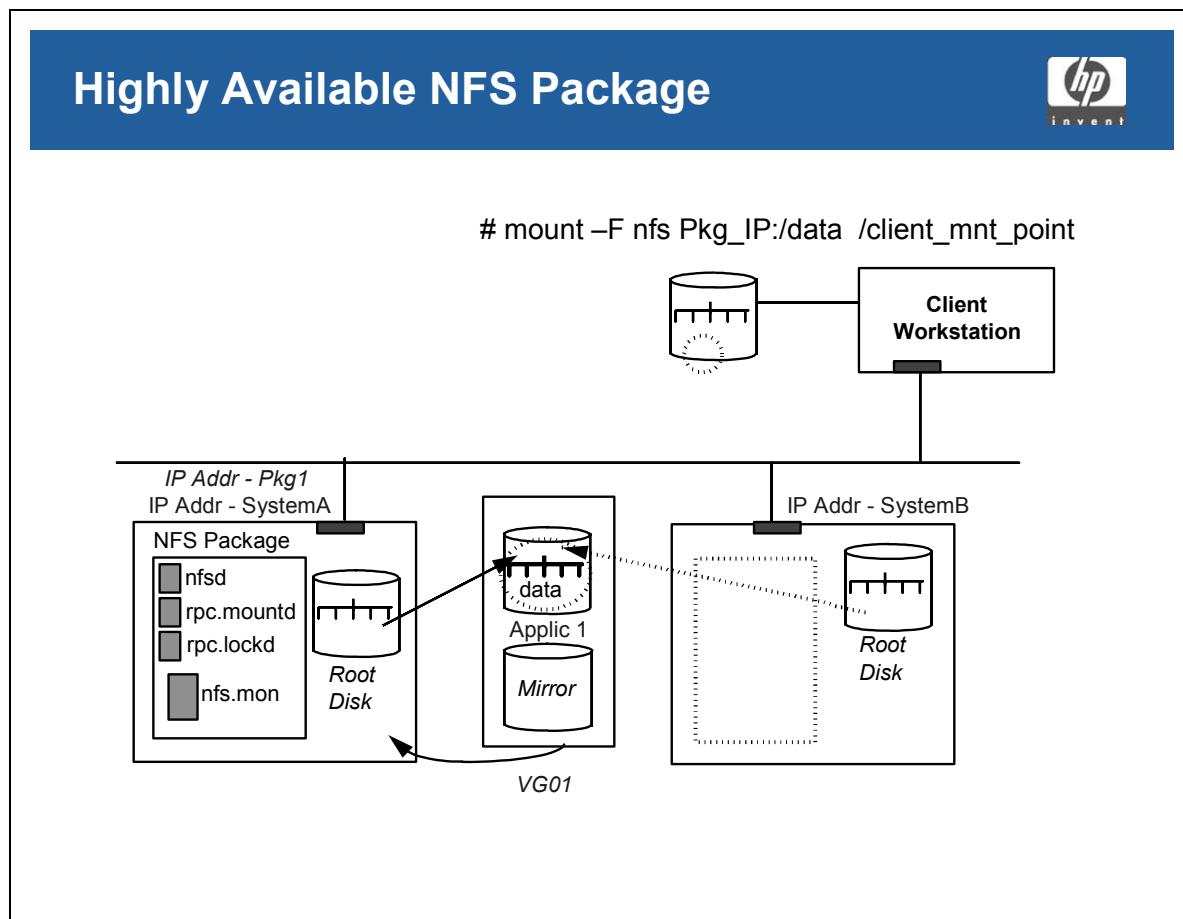
## **Module 12 — Highly Available NFS**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- Use the Serviceguard NFS toolkit.
- List files in the NFS toolkit.
- Set up a Serviceguard package using the NFS toolkit.

## 12-1. SLIDE: Highly Available NFS Package



### Student Notes

The purpose of highly available NFS servers is to ensure that clients will be able to access files and data located on a server file system. This is most beneficial to file server-based applications and data servers.

The concept is to group the daemons related to NFS into a package that can run on multiple hosts. The NFS file system is then placed in a shared volume group which any host in the cluster can access. Once configured, any host in the cluster is capable of providing NFS access to the data for the client workstations on the network.

The clients access the NFS file system through the package IP address, not the IP address of a host. The mount command issued by a client workstation would be in the following form:

```
# mount -F nfs NFS_Package_IP:/NFS_exported_fs /local_mnt_point
```

If the node providing NFS access to the file system fails, Serviceguard will move the NFS package to another node within the cluster. The new node will activate the volume group, mount the file systems, check that the NFS daemons are running, execute the `customer_defined_run_cmds` and start the NFS monitor script.

Client NFS packets will be automatically routed to the new host because that is where the package IP address now resides. Clients will be able to continue to access their data without remounting.

## 12–2. SLIDE: NFS Toolkit Files

### NFS Toolkit Files



As of SG version 11.17, the NFS Toolkit (part number B5140BA) is available for HPUX 11.00, HP-UX 11.11, and HP-UX 11i v2.

The NFS Toolkit contains the following files:

- README
- hanfs.sh
- nfs.flm
- nfs.mon
- nfs\_xmnt

The `nfs.conf` and `nfs.cntl` files are needed, as well, and are created with the `cmmakepkg` command.

### Student Notes

The NFS toolkit is made up of five files. The files are located in the `/opt/cmcluster/nfs` directory and are typically copied from there to the `/etc/cmcluster/nfs` directory.

The following is a listing of these five files, and a very brief description of each file.

README	A general discussion of the toolkit.
hanfs.sh	This script was new at version 11.14 of Serviceguard and the NFS Toolkit. This script is where we will identify the file system which will be nfs-exported. In this file, we also identify the <code>NFS_SERVICE_NAME</code> , <code>NFS_SERVICE_CMD</code> and the <code>NFS_SERVICE_RESTART</code> values.
nfs.flm	This script handles the NFS file lock propagation on HA servers cluster-wide. It contains specific environment variables and functions for providing the NFS file lock migration feature.
nfs.mon	The NFS monitoring script. This will be the service process that monitors the health of the NFS daemons.
nfs_xmnt	A customized script used for cross-mounting the NFS Server package.

## 12-3. TEXT PAGE: The nfs.flm ( “File lock migration” ) Script

The following is a copy of the nfs.flm “File lock migration” script for HP-UX 11i v2.

```
#!/usr/bin/sh
# A.11.23.02
# ****
# *      *
# *      SERVICEGUARD NFS TOOLKIT FILE LOCK MIGRATION SCRIPT      *
# *      *
# *      Note: This is a template NFS file lock migration script.      *
# *      It MUST be edited before it can be used.      *
# *      *
# ****
#####
# This script handles the NFS file lock propagation on HA servers
# cluster wide. It contains specific environment variables and functions
# for providing the NFS file lock migration feature.
#
# Each HA/NFS package requiring NFS File Lock Migration after a failover
# should run an invocation of this script and specify a unique holding
# directory via the NFS_FLM_HOLDING_DIR variable.
#
# Implementation:
#
# Use cp(1) to copy the contents of the /var/statmon/sm directory to
# a holding directory in one of the disk volumes associated with
# this specific HA/NFS package.
#
# During package start-up, the contents of the holding directory
# will be merged into the /var/statmon/sm directory on the server
# running the package.
#
#####
# The NFS_FLM_HOLDING_DIR variable defines the holding directory in a
# disk volume associated with the HA/NFS package. This directory
# will hold copies of the files from the /var/statmon/sm directory
# and will migrate with the HA/NFS package during a failover. The
# contents of this directory will be used to populate the
# /var/statmon/sm directory when recovering from a failover event.
#
# Select a directory in a file system that is part of the
# HA/NFS package.
#
# Example: NFS_FLM_HOLDING_DIR="/export/sm"
#
# where "/export/sm" is a directory residing in a disk volume
# associated with this HA/NFS package.
#
NFS_FLM_HOLDING_DIR=""

#
# Define the interval in seconds to copy the file lock state
# (i.e. monitoring entries) files into the holding directory.
# The default is 5 seconds.
#
```

## Module 12

### Highly Available NFS

```
PROPAGATE_INTERVAL=5

function validate_holding_dir {
#
# Make sure the configured holding directory (configured via the
# NFS_FLM_HOLDING_DIR variable) exists and has the proper ownership
# and permissions.
#
# First check to make sure the NFS_FLM_HOLDING_DIR variable has
# been configured. If not, print an error and exit.
#
if [[ -z $NFS_FLM_HOLDING_DIR ]]
then
    print "HANFS FLM -- $(date '+%b %e %X') - Node \c"
    print "\"$hostname\" : The NFS File Lock Migration holding \c"
    print "directory has not been configured."
    print "HANFS FLM -- $(date '+%b %e %X') - Node \c"
    print "\"$hostname\" : Set the NFS_FLM_HOLDING_DIR variable \c"
    print "in the ${0##*/} script. Exiting."
    exit 1
fi

if [[ ! -d $NFS_FLM_HOLDING_DIR ]]
then
    print "HANFS FLM -- $(date '+%b %e %X') - Node \c"
    print "\"$hostname\" : The NFS File Lock Migration holding \c"
    print "directory \"$NFS_FLM_HOLDING_DIR\" does not exist."
    print "HANFS FLM -- $(date '+%b %e %X') - Node \c"
    print "\"$hostname\" : Creating directory \"$NFS_FLM_HOLDING_DIR\""

    mkdir -p -m 644 $NFS_FLM_HOLDING_DIR 2>&1
    if [[ $? -ne 0 ]]
    then
        print "HANFS FLM -- $(date '+%b %e %X') - Node \c"
        print "\"$hostname\" : Failed to create \c"
        print "\"$NFS_FLM_HOLDING_DIR\". Exiting."
        exit 1
    fi

    chown root:sys $NFS_FLM_HOLDING_DIR
else
    chmod 644 $NFS_FLM_HOLDING_DIR
    chown root:sys $NFS_FLM_HOLDING_DIR
fi
}

function copy_sm_files {
#
# This function is used to copy Status Monitor files between
# the package holding directory (configured via the
# NFS_FLM_HOLDING_DIR variable) and the Status Monitor
# directory (/var/statmon/sm). This function takes two input
# parameters - the first is the source directory to copy SM
# files from and the second is the target directory to copy
# SM files to.
#
# The server's Status Monitor directory (/var/statmon/sm) gets
# populated just prior to killing and restarting the
# rpc.statd and rpc.lockd daemons during package startup.
# The holding directory get populated with current SM entries
# at a defined interval (PROPAGATE_INTERVAL) while the package
```

```

# runs. This allows the holding directory to only maintain
# the list of NFS clients that have performed file locks since
# the lockd/statd daemons were re-started at package start time.
#
# Before copying files, check to see if there are any files in
# the source directory to copy. If there are then copy the
# files and check for errors that occurred during the copy.
# Only print notification of successful file copy in the case
# where we are populating the /var/statmon/sm directory.
#
SOURCE_DIR=$1
TARGET_DIR=$2

if [ -f $SOURCE_DIR/* ]
then
    cp $SOURCE_DIR/* $TARGET_DIR/. 2>&1
    if [[ $? -ne 0 ]]
    then
        print "HANFS FLM -- $(date '+%b %e %X') - Node \c"
        print "\\"$hostname\\": FAILED to copy NFS File Lock \c"
        print "entries from \"$SOURCE_DIR\" to \"$TARGET_DIR\""
        exit 1
    else
        if [[ $SOURCE_DIR = $NFS_FLM_HOLDING_DIR ]]
        then
            print "HANFS FLM -- $(date '+%b %e %X') - Node \c"
            print "\\"$hostname\\": Populated \c"
            print "\\"$TARGET_DIR\\" with contents of \c"
            print "\\"$SOURCE_DIR\\": \c"
            ls $SOURCE_DIR 2>&1
        fi
    fi
fi
}

function remove_stale_sm_files {
#
# This function simply removes any files residing in the holding
# directory. It is called only after the holding directory contents
# have been copied to the SM directory during package startup.
#
if [ -f $NFS_FLM_HOLDING_DIR/* ]
then
    rm -f $NFS_FLM_HOLDING_DIR/* 2>&1
    if [[ $? -ne 0 ]]
    then
        print "HANFS FLM -- $(date '+%b %e %X') - Node \c"
        print "\\"$hostname\\": Failed to remove stale FLM \c"
        print "entries from holding directory"
    else
        print "HANFS FLM -- $(date '+%b %e %X') - Node \c"
        print "\\"$hostname\\": Removed stale FLM entries \c"
        print "from \"$NFS_FLM_HOLDING_DIR\" directory"
    fi
fi
}

```

## Module 12

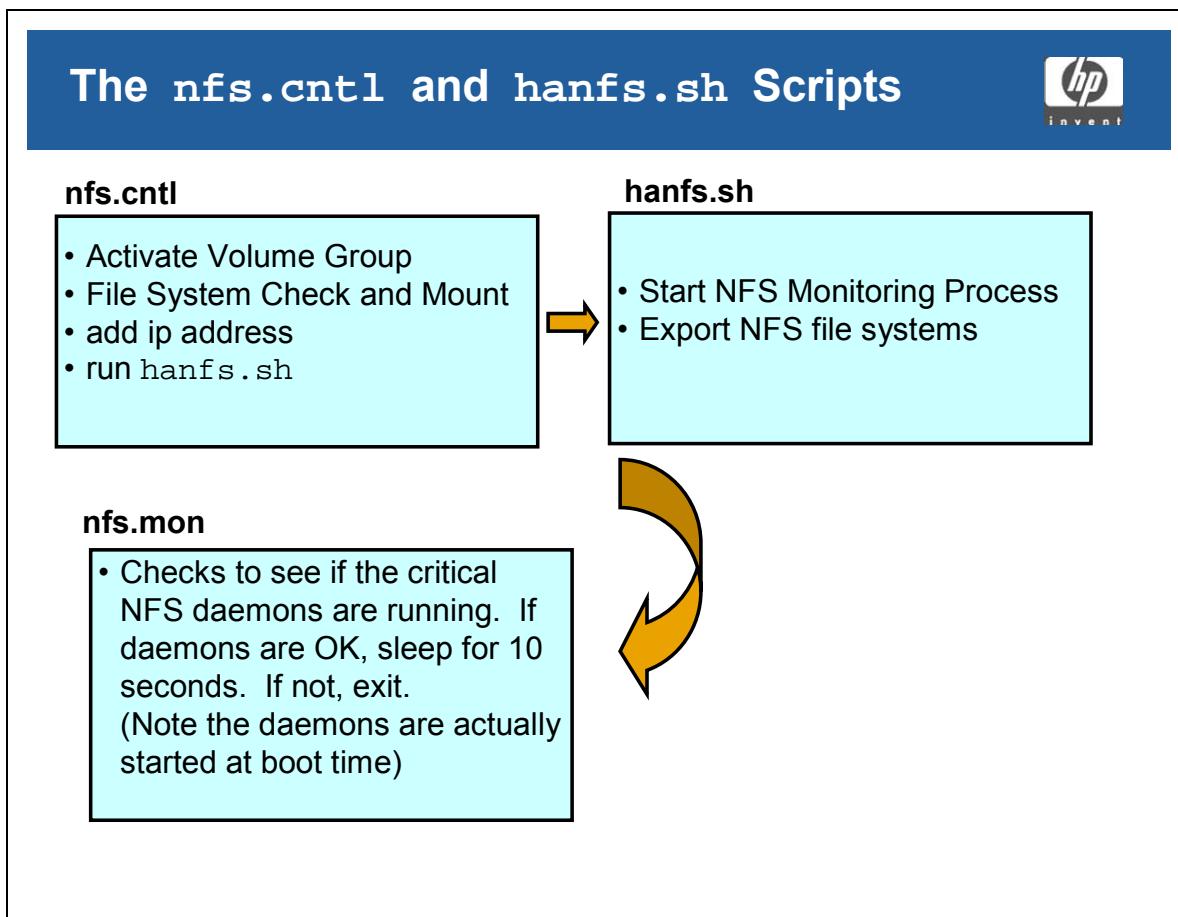
### Highly Available NFS

```
#-----MAINLINE NFS Code Starts Here-----
#
# Make sure the holding directory exists and has the proper permissions
#
validate_holding_dir

#
# Test to see if we are being called to populate the sm directory
# or populate the holding directory.
#
if [[ $1 = "populate_sm_dir" ]]
then
    copy_sm_files $NFS_FLM_HOLDING_DIR /var/statmon/sm
    remove_stale_sm_files
elif [[ $1 = "populate_holding_dir" ]]
then
    while true
    do
        copy_sm_files /var/statmon/sm $NFS_FLM_HOLDING_DIR
        sleep $PROPAGATE_INTERVAL
    done
fi
```

```
*** End of nfs.flm script ***
```

## 12-4. SLIDE: The nfscntl and hanfs.sh Scripts



### Student Notes

The slide shows the major activities performed by the NFS package.

- Activates the volume group or volume groups associated with the package.
- Mounts each file system associated with the package.
- Initiates the NFS monitor script to check periodically on the health of NFS services, if you have configured your NFS package to use the monitor script.
- Exports each file system associated with the package so that it can later be NFS-mounted by clients.
- Assigns a package IP address to the LAN card on the current node.

After this sequence, the NFS server is active, and clients can NFS-mount the exported file systems associated with the package.

## 12–5. SLIDE: Using the NFS Toolkit

### Using the NFS Toolkit



1. Ensure the `/etc/rc.config.d/nfsconf` file on all nodes is configured to start the NFS daemons
2. Create a directory for the new nfs package
3. Copy the files from the toolkit to the nfs package directory
4. Run `cmmakepkg` to create the `nfs.conf` and `nfs.cntl` files
5. Edit `nfs.conf`
  - use `nfs.service` as the `SERVICE_NAME`
6. Edit `nfs.cntl`
  - specify volume group, file system names
  - supply package IP/subnet
7. Edit the `hanfs.sh` file
  - Specify which file systems are to be exported
8. Run `cmcheckconf` and `cmaplyconf` to add the package

### Student Notes

Follow these procedures to configure Highly Available NFS

1. On the primary node and all adoptive nodes for the NFS package, set the `NFS_SERVER` and `START_MOUNTD` variables to 1 in the `/etc/rc.config.d/nfsconf` file.

Do not configure the exported directories in the `/etc/exports` file.

2. Use the `mkdir` command to create a directory for the nfs package. In our example we'll call the directory `/etc/cmcluster/nfs`:

```
# mkdir /etc/cmcluster/nfs
```

3. Copy the files from the NFS toolkit directory to the new NFS package directory:

```
# cp /opt/cmcluster/nfs/* /etc/cmcluster/nfs
```

4. Use `cmmakepkg` to create the package control file and run/halt script:

```
# cd /etc/cmcluster/nfs  
  
# cmmakepkg -p nfs.conf  
  
# cmmakepkg -s nfs.cntl
```

5. Make the following changes to the `nfs.conf` file created by `cmmakepkg`:

PACKAGE_NAME	<name of your nfs package>
NODE_NAME	<primary node name>
NODE_NAME	<alternate node name>
RUN_SCRIPT	/etc/cmcluster/nfs/nfs.cntl
HALT_SCRIPT	/etc/cmcluster/nfs/nfs.cntl
SERVICE_NAME	nfs_service
SUBNET	<subnet to be monitored>

6. Make the following changes to the `nfs.cntl` file created by `cmmakepkg`:

```
VG[0]=<vgname>  
LV[0]=<lvname>; FS[0]=<mount_point>; FS_MOUNT_OPT[0]=<options>  
IP[0]=<pkg ip address>  
SUBNET[0]=<subnet address>
```

Note, if you wish to have more than one highly available nfs package you can use the variable `HA_NFS_SCRIPT_EXTENSION=<suffix to append to hanfs.sh>`. For example, in your first nfs package you might set the variable to `pkg1.sh`. The `hanfs.sh` script for this package would then be called `hanfs.pkg1.sh`.

7. Edit the `hanfs.sh` script:

```
XFS[0]=<mount point of file system to export>  
NFS_SERVICE_NAME[0]="nfs.service"  
NFS_SERVICE_CMD[0]="/etc/cmcluster/nfs/nfs.mon"  
NFS_SERVICE_RESTART[0]="-r 0"
```

You do not have to run the NFS monitor script. If your NFS package configuration file specifies `AUTO_RUN YES` and `LOCAL_LAN_FAILOVER_ALLOWED YES` (the defaults), the package switches to the next adoptive node or to a standby network interface in the event of a node or network failure. The NFS monitor script causes the package failover if any of the monitored NFS services fails. If you do not want to run the NFS monitor script, comment out the `NFS_SERVICE_NAME` and `NFS_SERVICE_CMD` variables

8. Copy the files in the `/etc/cmcluster/nfs` directory to the other nodes in the cluster. This can be done with `rcp` or `ftp`. Make sure all the scripts are executable after they have been copied.

9. Run `cmcheckconf` and `cmapplyconf` to update the cluster binary.

## 12–6. SLIDE: NFS Script Variables

### NFS Script Variables



The “`hanfs.sh`” script contains several important variables:

- `XFS[ ]` These variables allow exported file systems to be defined and managed through the package control script instead of using the `/etc(exports` file.

Example: `XFS[0]="/data"`

Example: `XFS[1]="/tools"`

- `NFS_SERVICE_NAME` These variables define the service `nfs.mon`.
- `NFS_SERVICE_CMD` Because of these variables, no other services need to be defined through the standard SERVICE variables.
- `NFS_SERVICE_RESTART`

Note: Remember to uncomment the above lines.

### Student Notes

A number of shell variables tailored specifically for the NFS application are included in the `hanfs.sh` package script.

`XFS[0]` These variables specify the names of the highly available NFS file systems. These file systems are exported when the control script starts the package, and unexported when the control script halts the package. If NFS export options are needed, you may want to add an array for those options. For example if you wish to export with the read only option use `XFS[0]=" -o ro /sgdata1"`.

`NFS_SERVICE_NAME` This variable, along with `NFS_SERVICE_CMD` and `NFS_SERVICE_RESTART`, defines the `nfs.mon` monitoring script as a service process. If an additional service process is needed (not expected), it would be defined using the standard `SERVICE_NAME` variable.

---

## 12-7. LAB: The nfs Package

### Directions

Configure the cluster to contain a highly available NFS package. Use the NFS toolkit when configuring the package.

---

**NOTE:** The volume group names, disk device files, and volume group minor numbers must be localized for your system. This example uses an IP address and subnet. Your instructor will supply you with the correct ones to use.

---

### Part 1. Create a New Volume Group

1. First, on your cluster's primary node, and using two disks, create a new shared volume group named `vg02`, with a 200-MB, single-mirrored, logical volume named `sglvol2`, and build a `vxfs` file system in the logical volume. Then mount the file system at mount point `/sgdata2` and copy some files into it.

**Answer:**

```
# pvcreate [ -f ] /dev/rdsk/c_t_d_
# pvcreate [ -f ] /dev/rdsk/c_t_d_
# mkdir /dev/vg02
# mknod /dev/vg02/group c 64 0x020000
# vgcreate vg02 /dev/dsk/c_t_d0 /dev/dsk/c_t_d0
# lvcreate -L 200 -m 1 -n sglvol2 vg02
# newfs -F vxfs /dev/vg02/rsglvol2
# mkdir /sgdata2
# mount -F vxfs /dev/vg02/sglvol2 /sgdata2
# cp /sbin/lv* /sgdata2
```

2. Now, import this volume group to the other node in your cluster.

**Answer:**

**On the primary node:**

```
# vgexport -p -s -m /tmp/vg02.map vg02
```

**On the adoptive node:**

```
# cd /tmp
# rcp <primary_node>:/tmp/vg02.map .
# mkdir /dev/vg02
# mknod /dev/vg02/group c 64 0x020000
# vgimport -s -m /tmp/vg02.map vg02
# mkdir /sgdata2
# vgchange -a r vg02
# vgcfgbackup vg02
# vgchange -a n vg02
```

Module 12  
**Highly Available NFS**

3. Back on the primary node, unmount the file system and deactivate the volume group.

**Answer:**

```
# umount /sgdata2
# vgchange -a n vg02
```

4. While still on the primary node, mark the vg02 volume group as belonging to the Serviceguard cluster.

**Answer:**

```
# vgchange -c y vg02
```

## **Part 2. Configure the NFS Package**

1. Ask your instructor for an IP address. Assign your NFS package that assigned IP address. Update the `/etc/hosts` files on **both systems** with the package name and IP address.

**Answer:**

```
# vi /etc/hosts
```

2. On your primary node, in the `/etc/cmcluster` directory, create a directory called `nfs` for the NFS package named `nfs`. Copy the NFS toolkit files from `/opt/cmcluster/nfs` to your new directory.

**Answer:**

```
# mkdir /etc/cmcluster/nfs
# cd /etc/cmcluster/nfs
# cp /opt/cmcluster/nfs/* .
```

3. Use `cmmakepkg` to create the package configuration file.

**Answer:**

```
# cd /etc/cmcluster/nfs
# cmmakepkg -v -p nfs.conf
```

4. Edit `nfs.conf` as follows:

```
# vi nfs.conf
```

PACKAGE_NAME	nfs
NODE_NAME	<node1>
NODE_NAME	<node2>
RUN_SCRIPT	/etc/cmcluster/nfs/nfs.cntl
HALT_SCRIPT	/etc/cmcluster/nfs/nfs.cntl
SERVICE_NAME	nfs_service
SUBNET	10.x.x.x ← Use netstat -in to obtain 10-net subnet

5. Using `cmmakepkg`, create the package control script named `nfs.cntl`.

**Answer:**

```
# cmmakepkg -v -s nfs.cntl
```

6. Edit the `nfs.cntl` control script to customize it for your volume group, mount point, and the IP address you were assigned for this package.

```
# vi nfs.cntl
```

and make the following changes:

```
VG[ 0 ]="vg02"
```

In the logical volumes and file systems section, break up this line into separate lines, as we did in Module 10, and make the following changes:

LV[ 0 ]="/dev/vg02/sg1vol2"	← Note this value
FS[ 0 ]="/sgdata2"	← Note this value
FS_MOUNT_OPT[ 0 ]=" "	← Leave this value undefined
FS_UNMOUNT_OPT[ 0 ]=" "	← Leave this value undefined
FS_FSCK_OPT[ 0 ]=" "	← Leave this value undefined
FS_TYPE[ 0 ]="vxfs"	← Note this value

Module 12  
**Highly Available NFS**

Finally, in the IP addresses section, make the following changes:

IP[ 0 ]=10.x.x.x	← Use your assigned 10-net IP address here
SUBNET[ 0 ]=10.x.x.0	← Use netstat -in to obtain 10-net subnet

7. The last file to edit is the hanfs.sh file. Edit this file and make the following changes:
  - a. First, please go to line 67, which currently reads ...

```
#XFS[ 0 ]= "
```

and change this line to now read ...

```
XFS[ 0 ]="/sgdata2"           ← Do not forget to uncomment this line
```

- b. Next , go to line 99 and following, which now reads ...

```
#NFS_SERVICE_NAME[ 0 ]= ""
#NFS_SERVICE_CMD[ 0 ]= ""
#NFS_SERVICE_RESTART[ 0 ]= ""
```

and change these lines to read ...

```
NFS_SERVICE_NAME[ 0 ]="nfs_service"           ← Do not forget
NFS_SERVICE_CMD[ 0 ]="/etc/cmcluster/nfs/nfs.mon"   ← to uncomment
NFS_SERVICE_RESTART[ 0 ]="-r 0"                 ← these lines.
```

- c. Save your work and exit vi.
8. Ensure that the scripts in /etc/cmcluster/nfs are executable.

**Answer:**

```
# chmod 744 nfs.*
```

9. Next, create the /etc/cmcluster/nfs directory on the other system. Then, copy all these NFS files to the nfs subdirectory on the other system.

**Answer:**

```
# remsh <node2> "/usr/bin/mkdir /etc/cmcluster/nfs"
# rcp * <node2>:/etc/cmcluster/nfs/.
```

10. Now, back on your cluster's primary node, add your newly created volume group to your cluster-wide ASCII file.

**Answer:**

```
# cd /etc/cmcluster  
# vi cmclconfig.ascii
```

11. Execute `cmcheckconf` against both your NFS package conf file and the modified cluster-wide ASCII configuration file.

**Answer:**

```
# cd /etc/cmcluster/nfs  
# cmcheckconf -v -P nfs.conf -C ../cmclconfig.ascii
```

12. Update the cluster binary. You must do a `cmaplyconf` to set the volume group for Serviceguard exclusive activation.

**Answer:**

```
# cmaplyconf -v -P nfs.conf -C ../cmclconfig.ascii
```

13. Start the NFS package.

**Answer:**

```
# cmmodpkg -v -e nfs
```

14. Verify that the NFS package volume group is active, and the `/sgdata2` file system is mounted and exported on the node where the `nfs` package is running. If not, look at the package log file, correct any errors, and try again.

**Answer:**

```
# vgdisplay vg02  
# bdf  
# exportfs -v
```

Module 12  
**Highly Available NFS**

15. At this point, it is very helpful to perform several failovers. We want to make **absolutely sure** that the NFS package is behaving properly. Use a series of commands to test your failover such as ...
  - a. Kill the `nfs.mon` process. The package should fail over. Re-enable the node.
  - b. On the other node (where the `nfs` package should now be running), kill the `rpc.mountd` daemon. Do a `tail -f` on the `/etc/cmcluster/nfs/nfs.cntl.log` file on the node where the package was running. You should eventually be able to see the monitor attempting to contact the `rpc.mountd` daemon. Eventually the package should failover. Restart `rpc.mountd` by issuing the command: `/usr/sbin/rpc.mountd`. Re-enable the node
  - c. Halt, then restart the package

When you are convinced that the package is running and failing over properly, continue with step 16.

16. Move to a spare machine (*or use whatever machine your instructor suggests – this machine must be on the 10.x.x.x network*) and create a directory called `/your_name_nfstest` to be used as an NFS mount point.

**Answer:**

```
# mkdir /"your_name"_nfstest
```

17. Now, on this same spare machine, mount the file system from the NFS package

**Answer:**

```
# mount -F nfs <ip_address_of_package>:/sgdata2 /"your_name"_nfstest
```

18. Continuing on the spare machine, open a terminal window and write the following short "while true" loop to simulate access to the NFS package file system.

```
# while true
> do
> clear
> sleep 1
> ll /your_name_nfstest
> sleep 1
> done
```

19. Next, return to your Serviceguard cluster, and on the node on which your `nfs` package is running, and while the above "while true" loop is still running, kill the `nfs.mon` monitoring script. However, to understand this simulation, read the following paragraph before killing the `nfs.mon` process.

After you kill the `nfs.mon` process, watch carefully what happens to the "while" loop on the spare machine. You should see the "11" command hang (in the "while true" function in step 18 above), and then, depending on hardware, the failover should normally require less than 10 seconds. After the package fails over, you should see the "while true" loop pick up again with continuing access to the NFS-mounted file system!

**Answer:**

```
# ps -ef | grep nfs.mon      ← On the machine on which the package is running  
# kill <pid>                ← On the machine on which the package is running
```

20. Re-enable the switches for failover, and then try the package failover a few more times. Each time, please watch what happens to the "while true" loop on the spare machine.

**Answer:**

```
# cmmmodpkg -v -e -n <failed_node> nfs  
# ps -ef | grep nfs.mon    ← On the machine on which the package is running  
# kill <pid>                ← On the machine on which the package is running
```

21. Congratulations on being successful with your first package that uses all three of the following major components of a package:

- Disk resources
  - Network resources
  - Service Processes

## **12-8. LAB: NFS Package Performance**

### **Directions**

In this lab, we modify the NFS package so that the package is able to utilize as much of various system resources as possible. You can do this with two approaches:

- **Approach 1:** Shut down some other noncritical processes running on the system, or
- **Approach 2:** Shut down another package and move the specified package to another system automatically.

### **Approach 1: Shut Down Noncritical Processes (sendmail)**

#### **Special Note**

In the following example, we kill the `sendmail` process. Of course, `sendmail` has nothing to do with Serviceguard, but we use it here to demonstrate how non-critical processes on a node can be shut down when a package fails over to that node. This would be useful where the `NFS` package is mission critical, and must have as much CPU time as it needs.

1. Halt the nfs package and edit its control script as follows:

```
# cmhaltpkg -v nfs
# vi /etc/cmcluster/nfs/nfs.cntl

function customer_defined_run_cmds
{
    LOG="/etc/cmcluster/nfs/nfs.cntl.log"
    #
    # shutdown sendmail using the rc script
    #
    /sbin/init.d/sendmail stop
    echo "Sendmail daemon stopped by nfs_server \c"          >> $LOG
    echo "package on $(hostname) at $(date)."                  >> $LOG
    test_return 51
}
```

```
function customer_defined_halt_cmds
{
    LOG="/etc/cmcluster/nfs/nfs.cntl.log"
    #
    # startup sendmail using the rc script
    #
    /sbin/init.d/sendmail start
    echo "Sendmail daemon started by nfs_server \c"          >> $LOG
    echo "package on $(hostname) at $(date)."                  >> $LOG
    test_return 52
}
```

2. Next, perform the steps to get the modified nfs package running again.

```
# rcp /etc/cmcluster/nfs/nfs.cntl <node2>:/etc/cmcluster/nfs
# cmmodpkg -v -e nfs
# cmviewcl -v
```

3. On the node where the nfs package is running, is sendmail also running? Remember that if our package is running properly, sendmail should not be running.

**Answer:**

```
# ps -ef | grep sendmail
```

Module 12  
**Highly Available NFS**

4. Move the `nfs` package to the other node.

**Answer:**

```
# cmhaltpkg -v nfs
# cmrunpkg -v -n <other_node> nfs
# cmmodpkg -v -e nfs
```

5. Is `sendmail` running on the node where the `nfs` package is running? Did `sendmail` come back on the other system? Check the NFS log file for the messages sent by the `echo` commands from our `run/halt` script.

**Answer:**

The `sendmail` package should not be running on the same node as the `nfs` package. The `sendmail` package should have returned to the other node.

**Approach 2: Shut Down the "test" Package When the "nfs" Package Moves to a Node Where the "test" Package is Currently Running**

**Note**

In the following example, we deliberately fail over the `test` package. We do this to free up as much CPU time as possible for the `nfs` package. This would be useful where the `nfs` package is mission critical but the `test` package is not.

This approach will require you to put some additional logic in the function `customer_defined_run_cmds` function to stop the `test` package.

The sample solution will reflect only the information needed to move the `test` package. If you have other commands in the function, that is acceptable also. The sample solution will use `node1` and `node2` as your node names. Of course, you would use your specific hostnames.

1. Halt the `nfs` package and edit the `nfs.cntl` script. Remember to insert the correct hostnames where the script calls for `node1` and `node2`.

```
# cmhaltpkg -v nfs
# vi /etc/cmcluster/nfs/nfs.cntl
```

```
function customer_defined_run_cmds
{
    # move test package to other system
    cmhaltpkg -v test
    case $(hostname) in
        node1) cmmodpkg -v -e -n <node2> test ;;
        node2) cmmodpkg -v -e -n <node1> test ;;
    esac
    cmmodpkg -v -d -n $(hostname) test
    cmmodpkg -v -e test
    test_return 51
}
```

**Module 12**  
**Highly Available NFS**

2. Be sure to copy the modified control script to the other node in your cluster.

**Answer:**

```
# rcp /etc/cmcluster/nfs/nfs.cntl <node2>:/etc/cmcluster/nfs
```

3. Start the nfs package again.

**Answer:**

```
# cmmodpkg -v -e nfs
```

4. On the node where the nfs package is running, is the test package also running?

**Answer:**

```
# cmviewcl -v -p test
```

5. Move the nfs package to the other node.

**Answer:**

```
# cmhaltpkg -v nfs
# cmrunkpkg -v -n <other_node> nfs
# cmmodpkg -v -e nfs
```

6. Is the test package running on the node where the nfs package is running? Did the test package come back on the other system?

**Answer:**

```
# cmviewcl -v -p test
```

---

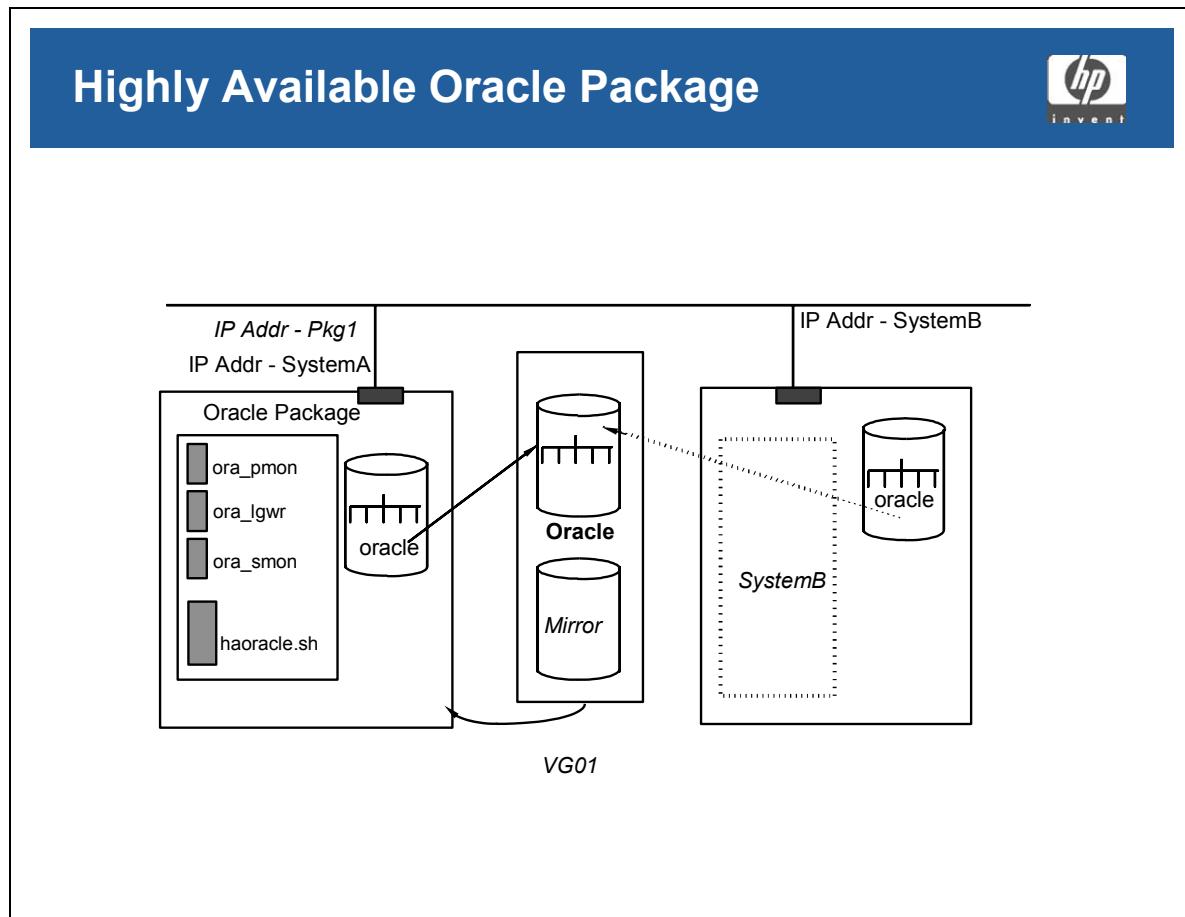
# **Module 13 — The Highly Available Oracle Database**

## **Objectives**

Upon completion of this module, you will be able to do the following:

- List different Serviceguard toolkits.
- Use Oracle toolkit and cookbook.
- Configure the Oracle 10g Database to run in a Serviceguard environment.

## 13-1. SLIDE: Highly Available Oracle Package – Overview



### Student Notes

One of the most common applications for a high availability environment is Oracle.

The following are recommendations for configuring Oracle for a Serviceguard environment:

- Install the Oracle product (libraries, binaries, commands) on all nodes of the cluster, as opposed to a single installation on a shared file system. By keeping the Oracle executable on each system rather than having one copy of the executables on a shared file system, there is less file system recovery (`fsck`) during a failover. This will help speed up the amount of time needed to failover the application.
- By having the executables on each system, you also can either patch or update the executable on one system and move the package to the updated system to test the new version of Oracle. However applying a major patch to the Oracle Software implies a ‘data dictionary’ change, which causes a full outage due to the fact that several scripts need to run against the database without user connected.
- Create a shared file system to hold the Oracle configuration files (`init.ora`, `spfile`, `listener.ora`, `tnsnames.ora`, `sqlnet.ora`).

- Create a shared file system to hold the archive log files.
- Create the Oracle database (including control files and redo logs) on a shared volume group. All clustered nodes will potentially need access to the database depending on where the package is located. A shared database with locally installed binaries is the recommended configuration.
- Oracle clients connect to the database via the package IP address, not the IP address of a host. Upon application failover, Oracle clients will need to reconnect using the same IP address. This is because Oracle uses TCP (as opposed to UDP), which is a connection-based protocol.
- An Oracle monitoring script will be needed to monitor the health of the Oracle application. HP offers an Oracle monitoring script as part of the database toolkit product.

## 13-2. SLIDE: Serviceguard Toolkits

### Serviceguard Toolkits



- Enterprise Cluster Master Toolkit (B5139DA)

Custom monitoring scripts for:

- Oracle
- Apache
- Samba
- Tomcat

### Student Notes

The toolkits shown on the slide have been developed to simplify the set up of packages related to NFS and a few other applications. Some applications cannot be started as service processes; therefore, they require an application monitoring script.

The primary value of the toolkits is the inclusion of an application monitoring script for each application. This eliminates the need for the customer to code a monitoring script from scratch.

The monitoring scripts included with the toolkits require minor modifications to customize the script for the customer's application.

The Enterprise Cluster Master Toolkit is intended to contain all future scripts for ease of ordering.

This toolkit is already loaded onto the system when one installs HP-UX 11i operating system with the "Mission Critical" operating environment. To see the toolkit files, look in the `/opt/cmcluster/toolkit/oracle` directory.

### 13-3. SLIDE: Database Toolkit Contents

## Database Toolkit Contents



### Oracle Toolkit Files

- README
- haoracle.mon and more
- General information and setup procedure
- Monitoring script for Oracle

### Apache Toolkit Files

- README
- hahttp.mon and more
- General information and setup procedure
- Monitoring script for Apache

### Samba Toolkit Files

- README
- hasmb.sh and more
- General information and setup procedure
- Monitoring script for Samba

### Tomcat Toolkit Files

- README
- hatomcat.sh and more
- General information and setup procedure
- Monitoring script for Tomcat

### Student Notes

There are four different database products for which toolkits currently exist.

The content of each toolkit is:

README file

This file contains configuration information for monitoring in a Serviceguard environment.

Monitoring script

This script is designed to monitor critical processes related to the specific application. This script must be edited for your specific application before it can be used. Other scripts are also included in the toolkit for each of the above applications.

## 13-4. SLIDE: Example: Oracle Application

### Example: Oracle Application



1. Activate Volume Group
2. File system Check and Mount
3. Add IP Address

4. Customer Defined Run Commands  
Start oracle database here



5. Start Oracle Monitoring Process (aka a service process)

haoracle.mon -checks to see if critical oracle daemons are running  
If daemons are OK, sleep for 10 seconds and check again.  
If daemons are NOT OK, try to fix. If cannot be fixed, exit.

### Student Notes

The slide shows the five major activities performed by the package control script when starting an Oracle database application.

1. Activate the volume group containing the Oracle database.
2. Check and mount any file systems related to the database.
3. Add the IP address for the Oracle package.
4. Execute any commands in the `customer_defined_run_cmds` function. This function is where commands to start the Oracle application are added.
5. Start the Oracle monitoring script to monitor the Oracle application started in step 4.

Before attempting to write your own monitoring script, check to see if a monitoring script already exists for your application.

## 13-5. SLIDE: Oracle Control Script Modification

### Oracle Control Script Modification



- SERVICE\_NAME[ 0 ]= Use these variables to call the haoracle.sh script with the monitor parameter.
- SERVICE\_CMD[ 0 ]=
- SERVICE\_RESTART[ 0 ]=
  
- customer\_defined\_run\_cmds{ Use this function to start the Oracle DB instance. Use the haoracle.sh script with the start parameter.  
haoracle.sh start  
}
- customer\_defined\_halt\_cmds{ Use this function to stop the Oracle DB instance. Use the haoracle.sh script with the shutdown parameter.  
haoracle.sh shutdown  
}

### Student Notes

The haoracle.sh script is used for *three* purposes. It is used to:

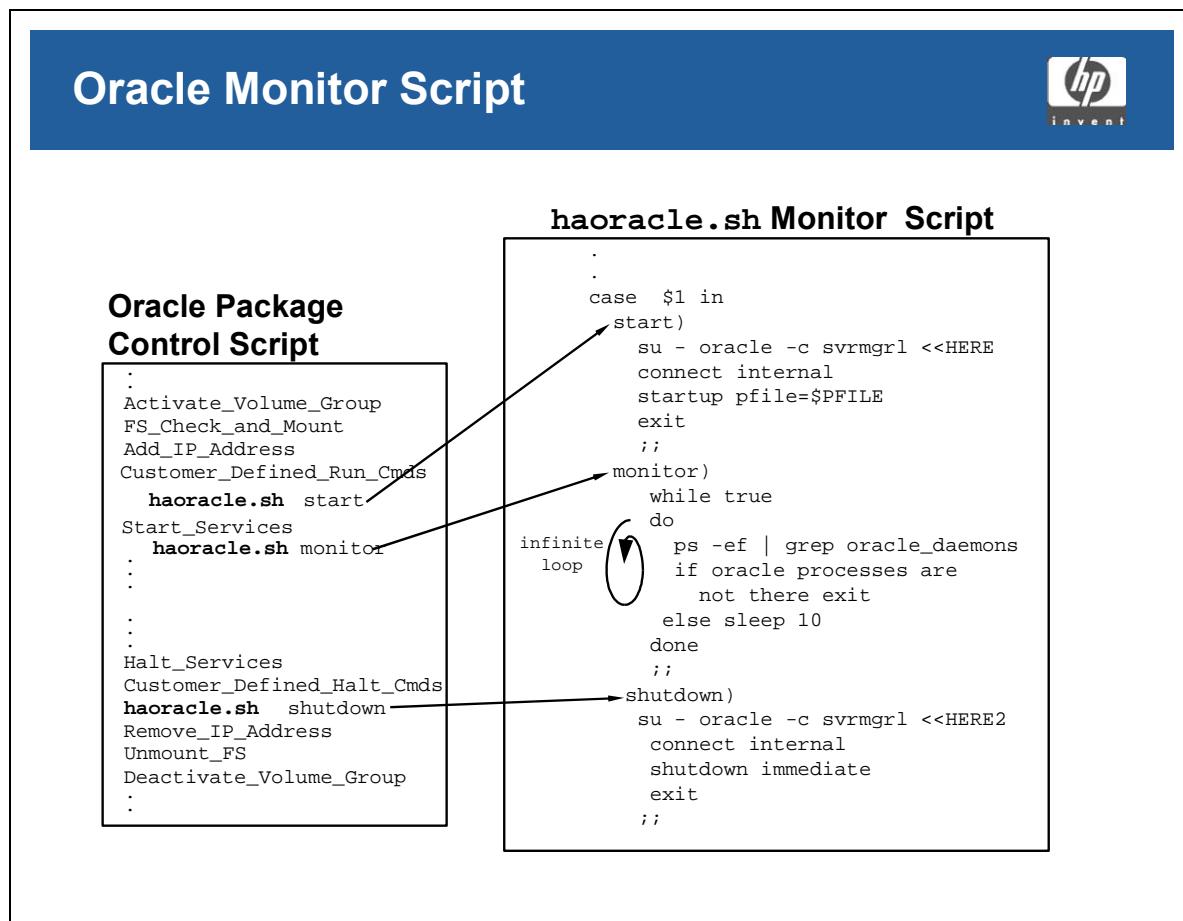
- Monitor the database (defined in the SERVICE\_CMD variable).
- Start the database (defined in the customer\_defined\_run\_cmds function).
- Bring down the database (defined in the customer\_defined\_halt\_cmds function).

```
SERVICE_NAME[ 0 ]=oracle_class
SERVICE_CMD[ 0 ]="/etc/cmcluster/oracle/haoracle.sh monitor"

function customer_defined_run_cmds {
    /etc/cmcluster/oracle/haoracle.sh start
    test_return 51
}

function customer_defined_halt_cmds {
    /etc/cmcluster/oracle/haoracle.sh shutdown
    test_return 52
}
```

## 13–6. SLIDE: Oracle Monitor Script



### Student Notes

The Oracle start/stop/monitoring script (`haoracle.sh`) provided in the database toolkit calls other scripts provided. With this toolkit version the start/stop/abort commands are located in `haoracle_sql.sh` and the monitoring is performed by `haoracle.mon`.

Moving the start up and shutdown of Oracle to the toolkit script guarantees the correct Oracle instance will be started/stopped and monitored.

This means the Oracle **control** script does the following:

- Activates the volume group(s)
- Checks and mounts the file systems
- Adds the package IP address
- Spawns the Oracle monitoring script as the service process

The Oracle **monitor** script monitors the Oracle database daemons.

## 13-7. SLIDE: Oracle Cookbook

### Oracle Cookbook



1. Create directory and shared LV for Oracle database specific files.
2. su - oracle
3. svrmgrl (Set up datafiles ,logfiles, database, tablespace)
4. mkdir /etc/cmcluster/oracle/
5. cd /etc/cmcluster/oracle
6. cmmakepkg -v -p oracle.conf and vi oracle.conf
7. cmmakepkg -v -s oracle.cntl and vi oracle.cntl
8. cp /opt/cmcluster/toolkit/oracle/haoracle.sh .
9. vi haoracle.sh
10. cmcheckconf and cmapplyconf and cmrunc1
11. Start the Oracle package and test database failover.

### Student Notes

The slide provides a high-level overview of the steps for configuring Oracle as a package within the Serviceguard environment.

#### Create the Subdirectory for the Oracle Instance (Step1)

Each Oracle instance must have its own subdirectory **structure** that contains all the configuration files, binaries, libraries, commands, and any other files that the **instance** might require.

Rather than reinstall these files for each instance, symbolically link to an already existing installation.

#### Create the Oracle Database, redo Logs, and Tablespace (Steps 2 and 3)

Once the instance files have been created (or linked), the database and other related files and tables need to be created. These items must be created by the Oracle account.

**Create the Package Configuration File and Control Script (Steps 4 - 7)**

To package the Oracle application for Serviceguard, a package configuration file and a package control script will need to be created using the standard tool, **cmmakepkg**. Both files will require customization related to the specific Oracle instance.

**Create the Oracle Start / Stop and Monitoring Script (Steps 8 and 9)**

The Oracle monitoring script that is included with the database toolkit product will need to be customized to be consistent with the specific Oracle instance.

The package control script and the Oracle monitoring script need to be copied to all nodes within the cluster.

**Compile and Distribute the Binary File (Step 10)**

The final steps are to rebuild the binary file using the standard **cmcheckconf** and **cmaplyconf** commands and then to bring up the cluster.

**Verify Cluster and Package Status (Step 11)**

The **cmviewcl** command verifies the Oracle package is up and running properly. If the Oracle package did not come up, the package control script log file, the monitor script log file, and the **syslog.log** files should be checked for errors.

## 13-8. LAB: Oracle Package Configuration

### Introduction

1. This lab will setup an Oracle database to work with Serviceguard.
2. The volume group names, disk device files, and volume group minor numbers must be localized for your system. This example uses IP addresses and subnets. Your instructor will supply you with the correct ones to use.
3. This lab is structured to minimize the Oracle knowledge you need to know. The lab creates a database using some shell scripts that have been placed on your system in the /labs/mod13 subdirectory.
4. This lab reinforces where the 3 components of setup are needed.
  - Basic system administration setup and network configuration
  - Creation of the Oracle database
  - Packaging the Oracle database to work with Serviceguard
5. These systems were setup with a user named "oracle". The Oracle executables are loaded in the /oral/oracle/product/10.1.0.db directory on **both systems**.
6. One objective of the lab is to create an environment where several instances of Oracle could run concurrently. This would allow one instance to move while leaving the other Oracle instances alone.
7. A second objective is to share the Oracle executables among all instances on a system. Each instance will be symbolically linked to the binaries under the subdirectory ...

/opt/oracle/product/10.1.0.db

A third and final objective is to allow each instance to run on any node in the cluster. To do this, each instance will need to be configured on its own volume group and file system.

---

**NOTE:** This course creates *one* Oracle instance. Creating multiple Oracle instances with Serviceguard is covered in "Disaster Tolerant Architectures with Serviceguard, Oracle 9i and Oracle 10gRAC" course, HP course number U8601S.

---

---

**NOTE:** In this course, due to the manner in which the lab setup was done, and the hard-coding of some variables, only *one* Oracle package is possible per Serviceguard cluster.

---

### Special Notes for the Whole Lab

1. In the following steps, `Node1` will always refer to the primary node for your Oracle package, and `Node2` will always refer to the adoptive or alternate node.
2. It is **vitally important** that you execute each step in this lab on the correct node!
3. Please complete all parts of this lab entirely on your cluster's primary node unless otherwise specifically indicated.
4. It is also **vitally important** that you execute each step as the correct user, either the `root` user or the `oracle` user. Pay special attention to this!

## Part 1: Prepare Oracle For an Oracle 10g Environment

### Important Special Note for Part 1 ONLY

When executing the script below, it is **critically important** to do so using an ASCII terminal. Remote users should establish a Putty (or similar) session to their system. Do not use ReflectionX (or any other graphical terminal emulator) as the execution time of the following script will be much longer. When you use an ASCII terminal, the script should execute in as little as 4 minutes. Using a GUI such as ReflectionX will mean that the execution time will be MUCH greater!

1. To ensure success later in this lab, execute the following script on **both nodes**. In an ASCII terminal, this script should require about 4 or 5 minutes to execute.

```
<Node1> # /labs/mod_13/oracle_10g.sh  
<Node2> # /labs/mod_13/oracle_10g.sh
```

While this script is executing, and in another window, continue with Part 2 of the lab.

## Part 2: Set Up the LVM and File System Structures for the Oracle Database

1. [Node1/root] We need to have two disks still "unused". Therefore, it may be necessary for you to first cmhaldtpkg one or more packages and/or vgexport one or more volume groups in order to accomplish this. So, after freeing up two disks (*if needed*), create a shared volume group named vg03.

```
# mkdir /dev/vg03
# mknod /dev/vg03/group c 64 0x030000
# pvcreate [ -f ] /dev/rdsk/cWtXd0
# pvcreate [ -f ] /dev/rdsk/cYtZd0
# vgcreate vg03 /dev/dsk/cWtXd0 /dev/dsk/cYtZd0
# vgchange -a n vg03
# vgchange -c y vg03
# vgchange -a e vg03
```

2. [Node1/root] Next, create and mirror the Oracle logical volume named oracle1. Also, note that the oracle1 logical volume ends with a numeral one, not the letter "ell".

```
# lvcreate -L 3000 -m 1 -n oracle1 vg03
```

3. [Node1/root] In the /dev/vg03/oracle1 logical volume, create a **JFS** file system. Also, create a mount point at /oracle1. Mount the oracle1 logical volume at the /oracle1 mount point. Again, note that the oracle1 mount point ends with a numeral *one*, not the letter "ell".

```
# newfs -F vxfs /dev/vg03/oracle1
# mkdir /oracle1
# mount -F vxfs /dev/vg03/oracle1 /oracle1
```

**The Highly Available Oracle Database**

4. [Node1/root] Create the following directory structure to hold the Oracle database by executing the following script:

```
# /labs/mod_13/dirbuild.sh
```

This script contains the following commands:

```
# mkdir /oracle1/admin
# mkdir /oracle1/admin/bdump
# mkdir /oracle1/admin/cdump
# mkdir /oracle1/admin/udump
# mkdir /oracle1/admin/pfile
# mkdir /oracle1/admin/network
# mkdir /oracle1/oraredo
# mkdir /oracle1/oradata
# mkdir /oracle1/oraarch
# chown -R oracle:dba /oracle1
# chmod -R 755 /oracle1
```

5. [Node1/root] Next, create a map file for your new `vg03` volume group and transfer the map file to <node2>:

```
# vgexport -v -p -s -m /tmp/vg03.map vg03
# rcp /tmp/vg03.map <node2>:/tmp/vg03.map
```

6. [Node2/root] Now, on <node2>, import the new `vg03` definition.

```
# mkdir /dev/vg03
# mknod /dev/vg03/group c 64 0x030000
# vgimport -v -s -m /tmp/vg03.map /dev/vg03
```

Note: If warnings appear regarding the volume group belonging to a different CPU ID, you can ignore them.

<pre># vgchange -a r vg03 # vgcfgbackup vg03 # vgchange -a n vg03</pre>	← Activate vg03 in read-only mode ← Create a backup copy of the LVM structures ← Deactivate volume group vg03
---	---

## Part 3: Create the Oracle Database

1. [Node1/root] Copy Oracle's .profile, the Oracle network file ( listener.ora ) and create the ORACLE\_BASE directory. Set the appropriate ownership.

```
# cp /labs/mod_13/profile.oracle /home/oracle/.profile
# cp /labs/mod_13/listener.ora /oracle1/admin/network
# mkdir -p /opt/oracle/product
# chown oracle:dba /opt/oracle/product
# chown oracle:dba /oracle1/admin/network
# chown oracle:dba /home/oracle/.profile
```

2. [Node2/root] Copy Oracle's .profile and create the ORACLE\_BASE directory.

```
# cp /labs/mod_13/profile.oracle /home/oracle/.profile
# mkdir -p /opt/oracle/product
# chown oracle:dba /opt/oracle/product
# chown oracle:dba /home/oracle/.profile
```

3. [Node1/oracle] Copy the default configuration file to the oracle1 instance directory and link the pre-installed Oracle software to the ORACLE\_HOME used by the database.

```
# su - oracle
$ cp /labs/mod_13/initORACLE1.ctl /oracle1/admin/pfile
$ ln -sf /ora1/oracle/product/10.1.0.db /opt/oracle/product
$ exit
```

4. [Node2/oracle] Link the pre-installed Oracle software to ORACLE\_HOME used by the database.

```
# su - oracle
$ ln -sf /ora1/oracle/product/10.1.0.db /opt/oracle/product
$ exit
```

5. [Node1/oracle] Back on **<node1>**, su to the `oracle` account, and verify that profile changes have taken effect. You can do this by checking the value of `ORACLE_SID`.

```
# su - oracle
$ echo $ORACLE_SID           ← Verify the ORACLE_SID variable is ORA1SID
```

### Important Special Notes for Step 6

1. Note that we do not exit the `oracle` user account. Therefore, while still logged in as the `oracle` user, please continue with step 6 below.
  2. Most importantly, when executing the `create_database.sh` script (see step 6 below), it is critically important to do so using an ASCII terminal. Remote users should use Putty to system (or a similar terminal emulator). Do NOT use ReflectionX (or any other graphical terminal emulator). When you use an ASCII terminal, the script should execute in a little as 22 minutes. Using a graphical terminal emulator will mean that the execution time will be measured in hours!
6. [Node1/oracle] Next, still on the primary node, and still as the oracle user, execute the `/labs/mod_13/create/create_database.sh` script. Due to the way Oracle creates views, by trying to remove the view before creating it, many errors occur because the view name does not exist. These error messages are to be expected and should be ignored. Allow 30-40 minutes for this script to complete its execution.

```
$ /labs/mod_13/create/create_database.sh
```

This `create_database.sh` script contains the following code:

```
#####
#
# Create Database
#
# Configuration parameters are set
# in DB.conf
#
# Last Modification: 21/03/2005
#
#####
cd /labs/mod_13/create
sqlplus "/ as sysdba" @00_Main.sql
```

The script simply calls an sql script. All configuration is made in the `DB.conf` file.

The 00\_Main.sql script contains the following code:

```
-- -----
-- Main script to create an Oracle 10g database
--
-- Last modification: 27/05/2005
--

@DB.conf

-- -----
-- Create spool directory (if not already created) and delete logfiles of
-- previous database creation
host mkdir -p &&SPOOLDIR
host rm &&SPOOLDIR/* 2>/dev/null
--
host echo "Start database creation &&DBNAME at " `date` > &&SPOOLDIR/time.log
--
spool &&SPOOLDIR/sql.log append
--
-- Create password file to allow remote access as sysdba
host $ORACLE_HOME/bin/orapwd file=&&ORACLE_BASE/admin/pfile/orapw&&ORACLE_SID
password=&&sysPWD force=y
host ls -l &&ORACLE_BASE/admin/pfile/orapw&&ORACLE_SID
--
-- Link password file to $ORACLE_HOME/dbs
host ln -sf &&ORACLE_BASE/admin/pfile/orapw&&ORACLE_SID $ORACLE_HOME/dbs
--
-- Create init.ora
host echo "spfile='&&SPFILE'" > &&PFILE
host cat &&PFILE
--
-- Link init.ora to $ORACLE_HOME/dbs
host ln -sf &&PFILE $ORACLE_HOME/dbs/init&&ORACLE_SID..ora
host ls -l $ORACLE_HOME/dbs
--
-- -----
create
  spfile='&&SPFILE'
  from
  pfile='&&PFILE_ASCII'
;
-- -----
host ./10_CreateDB.sh 3 2 2 2 &&REDOLOGS &&ASM
@@10_CreateDB.sql
--
@?/rdbms/admin/catalog.sql;
@?/rdbms/admin/catblock.sql;
@?/rdbms/admin/catproc.sql;
@?/rdbms/admin/catoctk.sql;
@?/rdbms/admin/owminst.plb;
@?/rdbms/admin/dbmspool.sql;
connect SYSTEM/&&systemPWD
@?/sqlplus/admin/pupbld.sql;
@?/sqlplus/admin/help/hlpbld.sql helpus.sql;
create table test (num number, name varchar2(15), city varchar2(20));
truncate table test;
insert into test values( 0, 'MJ','Chicago, IL');
insert into test values( 1, 'Magic','Los Angeles, CA');
insert into test values( 2, 'Bird','Boston, MA');
insert into test values( 3, 'Gary','Detroit, MI');
insert into test values( 4, 'Martin','Frankfurt, Germany');
```

## The Highly Available Oracle Database

```
insert into test values( 5, 'Vince','Toronto, Canada');
commit;
--
connect / as sysdba
@$ORACLE_HOME/javavm/install/initjvm.sql;
@$ORACLE_HOME/xdk/admin/initxml.sql;
@$ORACLE_HOME/xdk/admin/xmlja.sql;
@$ORACLE_HOME/rdbms/admin/catjava.sql;
@$ORACLE_HOME/rdbms/admin/catexf.sql;
--
host echo "End database creation &&DBNAME at " `date` >> &&SPOOLDIR/time.log
select * from system.test;
shutdown immediate
spool off
exit;
```

The DB.conf script contains the following code:

```
-- #####  
--  
-- Name: DB.conf  
--  
-- Author: Martin Lehmann  
--  
-- Creation date: 27/05/2005  
--  
-- Description: Oracle Environment and DB creation parameters  
--  
-- This file contains all parameters to run the DB-create scripts.  
-- It needs to be included in each script.  
--  
-- Directories:  
-- -----  
-- SPOOLDIR      : Script Output Directory  
-- ORACLE_BASE   : e.g. /opt/oracle  
-- ORACLE_HOME   : e.g. /oracle1/product/10.1.0  
--  
-- -----  
--  
-- Controfile Parameters:  
-- -----  
-- DBNAME        : Database Name (default service name)  
-- INSTANCES     : No. of instances, that will be setup for  
--                  this database.  
-- REDOLOGS       : No. of redolog files per thread  
-- REDOLOG_SIZE   : Size of each redolog  
-- MAXLOGFILES   : minimum: INSTANCES*REDOLOGS  
-- MAXLOGHISTORY : minimum: MAXLOGFILES*8  
-- MAXLOGMEMBERS : Max. No. of redo members in a single group  
-- MAXDATAFILES  : Max. No. of datafiles, that can be used for  
--                  the database  
-- CHARSET        : database default character set  
"DB.conf" [Read only] 124 lines, 3634 characters  
--  
define SPFILE=&&ORACLE_BASE/admin/pfile/spfile&&DBNAME..ora  
-- define SPFILE=+&&DG_SPFILE/spfile&&DBNAME..ora  
--  
define ULTRASEARCH_HOST=$NODE1  
define ULTRASEARCH_PORT=30005  
define ULTRASEARCH_SERVICE=ULTRA  
--  
set serveroutput on  
set linesize 200  
set trimspool on  
set timing on  
set time on  
define _EDITOR=vi  
-----
```

The 00\_Main.sql script calls one other script, 10\_CreateDB.sh.

Module 13  
**The Highly Available Oracle Database**

The 10\_CreateDB.sh script contains the following code:

```
# /usr/bin/ksh
# -----
#
# Script: 10_CreateDB.sh
#
# Last Modification: 29/03/2005
#
# Generate DB creaescript based on variables passed:
#
# $1: No. of system datafiles
# $2: No. of sysaux datafiles
# $3: No. of temp datafiles
# $4: No. of undo datafiles
# $5: No. of redo logfiles
# $6: ASM (empty)
#
# -----
SYS=$1
AUX=$2
TMP=$3
UND=$4
RDO=$5
ASM=$6
#
echo "shutdown abort" >> 10_CreateDB.sql
echo "set echo on" >> 10_CreateDB.sql
echo "startup nomount" >> 10_CreateDB.sql
echo "create database \"&&DBNAME\" " >> 10_CreateDB.sql
echo "controlfile reuse" >> 10_CreateDB.sql
echo "maxinstances &&MAXINSTANCES" >> 10_CreateDB.sql
echo "maxloghistory &&MAXLOGHISTORY" >> 10_CreateDB.sql
echo "maxlogfiles &&MAXLOGFILES" >> 10_CreateDB.sql
echo "maxlogmembers &&MAXLOGMEMBERS" >> 10_CreateDB.sql
echo "maxdatafiles &&MAXDATAFILES" >> 10_CreateDB.sql
echo "datafile" >> 10_CreateDB.sql
#
counter=0
i=${SYS}
k=','
while [ "${i}" != "${counter}" ]; do
    counter=`expr ${counter} + 1`
    if [ "${i}" = "${counter}" ]; then
        k=''
    fi
    if [ "${ASM}x" != "ASMX" ]; then # Filesystem or raw devices
        echo " '&&DATADIR/system0${counter}.dbf' size && \
            SYSTEM_SIZE reuse${k}" >>10_CreateDB.sql
    else # ASM
        echo " '+&&DG_ASM_SYSTEM' size &&SYSTEM_SIZE reuse${k}" \
            >>10_CreateDB.sql
    fi
done
#
echo " extent management local" >> 10_CreateDB.sql
echo "sysaux" >> 10_CreatedDB.sql
echo "datafile" >> 10_CreatedDB.sql
counter=0
i=${AUX}
k=','
while [ "${i}" != "${counter}" ]; do
```

```

counter=`expr ${counter} + 1`
if [ "${i}" = "${counter}" ]; then
    k=''
fi
if [ "${ASM}x" != "ASMx" ]; then # Filesystem or raw devices
    echo "  '&&DATADIR/sysaux${counter}.dbf' size &&SYSAUX_SIZE reuse${k}"
>>10_CreateDB.sql
else # ASM
    echo "  '+&&DG_ASM_SYSAUX' size &&SYSAUX_SIZE reuse${k}"
>>10_CreateDB.sql
    fi
done
#
echo "default temporary tablespace temp" >> 10_CreateDB.sql
echo "tempfile" >> 10_CreateDB.sql
counter=0
i=${TMP}
k=''
while [ "${i}" != "${counter}" ]; do
    counter=`expr ${counter} + 1`
    if [ "${i}" = "${counter}" ]; then
        k=''
    fi
    if [ "${ASM}x" != "ASMx" ]; then # Filesystem or raw devices
        echo "  '&&TEMPDIR/temp0${counter}.dbf' size &&TEMP_SIZE reuse${k}"
>>10_CreateDB.sql
    else # ASM
        echo "  '+&&DG_ASM_TEMP' size &&TEMP_SIZE reuse${k}" >>10_CreateDB.sql
    fi
done
#
echo 'undo tablespace "UNDOTBS1"' >> 10_CreateDB.sql
echo "datafile" >> 10_CreateDB.sql
counter=0
i=${UND}
k=''
while [ "${i}" != "${counter}" ]; do
    counter=`expr ${counter} + 1`
    if [ "${i}" = "${counter}" ]; then
        k=''
    fi
    if [ "${ASM}x" != "ASMx" ]; then # Filesystem or raw devices
        echo "  '&&UNDODIR/undotbs10${counter}.dbf' size &&UNDO_SIZE reuse${k}"
>>10_CreateDB.sql
    else # ASM
        echo "  '+&&DG_ASM_UNDO' size &&UNDO_SIZE reuse${k}" >>10_CreateDB.sql
    fi
done
echo "CHARACTER SET &&CHARSET" >> 10_CreateDB.sql
echo "NATIONAL CHARACTER SET &&NATIONAL_CHARSET" >> 10_CreateDB.sql
echo "LOGFILE" >> 10_CreateDB.sql
counter=0
i=${RDO}
k=''
while [ "${i}" != "${counter}" ]; do
    counter=`expr ${counter} + 1`
    if [ "${i}" = "${counter}" ]; then
        k=''
    fi
    if [ "${ASM}x" != "ASMx" ]; then # Filesystem or raw devices
        echo "  GROUP ${counter} ('&&REDODIR/redo0${counter}.log') SIZE
&&REDO_SIZE reuse${k}" >>10_Crea
teDB.sql
    fi
done

```

## Module 13

### The Highly Available Oracle Database

```
else # ASM
    echo " GROUP ${counter} ('+&&DG_ASM_REDO') SIZE &&REDO_SIZE reuse${k}"
>>10_CreateDB.sql
fi
done
echo 'USER SYS IDENTIFIED BY "&&SYSPWD"'
>>10_CreateDB.sql
echo 'USER SYSTEM IDENTIFIED BY "&&SYSTEMPWD";'
>>10_CreateDB.sql
```

The 10\_CreateDB.sh script creates a script 10\_CreateDB.sql depending on various parameters in DB.conf.

The following is an example of the output of 10\_CreateDB.sql:

```
shutdown abort
set echo on
startup nomount
create database "&&DBNAME"
controlfile reuse
maxinstances &&MAXINSTANCES
maxloghistory &&MAXLOGHISTORY
maxlogfiles &&MAXLOGFILES
maxlogmembers &&MAXLOGMEMBERS
maxdatafiles &&MAXDATAFILES
datafile
    '&&DATADIR/system01.dbf' size &&SYSTEM_SIZE reuse,
    '&&DATADIR/system02.dbf' size &&SYSTEM_SIZE reuse,
    '&&DATADIR/system03.dbf' size &&SYSTEM_SIZE reuse
    extent management local
sysaux
datafile
    '&&DATADIR/sysaux1.dbf' size &&SYSAUX_SIZE reuse,
    '&&DATADIR/sysaux2.dbf' size &&SYSAUX_SIZE reuse
default temporary tablespace temp
tempfile
    '&&TEMPDIR/temp01.dbf' size &&TEMP_SIZE reuse,
    '&&TEMPDIR/temp02.dbf' size &&TEMP_SIZE reuse
undo tablespace "UNDOTBS1"
datafile
    '&&UNDODIR/undotbs101.dbf' size &&UNDO_SIZE reuse,
    '&&UNDODIR/undotbs102.dbf' size &&UNDO_SIZE reuse
CHARACTER SET &&CHARSET
NATIONAL CHARACTER SET &&NATIONAL_CHARSET
LOGFILE
    GROUP 1 ('&&REDODIR/redo01.log') SIZE &&REDO_SIZE reuse,
    GROUP 2 ('&&REDODIR/redo02.log') SIZE &&REDO_SIZE reuse,
    GROUP 3 ('&&REDODIR/redo03.log') SIZE &&REDO_SIZE reuse,
    GROUP 4 ('&&REDODIR/redo04.log') SIZE &&REDO_SIZE reuse,
    GROUP 5 ('&&REDODIR/redo05.log') SIZE &&REDO_SIZE reuse
USER SYS IDENTIFIED BY "&&SYSPWD"
```

7. It is **absolutely** critical that your `create_database` script runs correctly. The last several lines of the script output should be the same as the listing below. If the last portion of your screen output does not look like the following, ask your instructor for assistance.

```
PL/SQL procedure successfully completed.

Elapsed: 00:00:07.41
11:00:38 SQL>
11:00:38 SQL> EXECUTE dbms_expfil_reg.validate_expfil;

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.09
11:00:38 SQL>
11:00:38 SQL> ALTER SESSION SET CURRENT_SCHEMA = SYS;

Session altered.

Elapsed: 00:00:00.00
11:00:38 SQL> --
11:00:38 SQL> host echo "End database creation &&DBNAME at " `date` >>
&&SPOOLDIR/time.log

11:00:38 SQL> select * from system.test;

      NUM NAME          CITY
----- -----
      0 MJ        Chicago, IL
      1 Magic     Los Angeles, CA
      2 Bird      Boston, MA
      3 Gary      Detroit, MI
      4 Martin    Frankfurt, Germany
      5 Vince     Toronto, Canada

6 rows selected.

Elapsed: 00:00:00.07
11:00:38 SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
11:01:08 SQL> spool off
11:01:08 SQL> exit;
Disconnected from Oracle Database 10g Enterprise Edition Release 10.1.0.2.0 -
64bit Production
With the Partitioning, Oracle Label Security, OLAP and Data Mining Scoring
Engine options

Fri Jun  3 11:01:09 EDT 2005
$
```

**The Highly Available Oracle Database**

8. [Node1/oracle] Once the script completes, exit the oracle user account and un-mount the Oracle file system. Then, de-activate the volume group vg03.

```
$ exit                                ← Exit the oracle account  
# umount /oracle1  
# vgchange -a n vg03
```

9. [Node2/root] Again on <node2>, create a mount point directory for the oracle1 package, and make it writable. Then activate the volume group and mount the file system /oracle1.

```
# mkdir /oracle1  
# chmod 777 /oracle1  
# vgchange -a e vg03  
# mount -F vxfs /dev/vg03/oracle1 /oracle1
```

10. [Node2/oracle] Create two required links from shared device to the local Oracle\_HOME/dbs directory.

```
# su - oracle  
$ ln -sf /oracle1/admin/pfile/orapwORALSID \  
          /opt/oracle/product/10.1.0.db/dbs  
$ ln -sf /oracle1/admin/pfile/initORALSID.ora \  
          /opt/oracle/product/10.1.0.db/dbs  
$ exit                                ← Exit the Oracle account
```

11. [Node2/root] Un-mount the oracle file system and de-activate volume group vg03.

```
# umount /oracle1  
# vgchange -a n vg03
```

12. [Node1/root] Create an entry in the `/etc/hosts` file to refer to a 10-net non-routable IP address that will be used by the Oracle Listener.

```
# vi /etc/hosts
```

and add the following line, where the `10.x.x.x` is replaced by a 10-net non-routable IP address provided by your instructor.

10.x.x.x	oralhost
----------	----------

When finished, exit the file and save your changes.

13. [Node2/root] Create an entry in the `/etc/hosts` file to refer to a 10-net non-routable IP address that will be used by the Oracle Listener.

```
# vi /etc/hosts
```

and add the following line, where the `10.x.x.x` is replaced by a 10-net non-routable IP address provided by your instructor.

10.x.x.x	oralhost
----------	----------

When finished, exit the file and save your changes.

## Part 4: Create the Package Configuration File and Control Script

1. **[Node1/root]** While still on <node1>, make a directory for the Oracle package files and cd into the new directory. Note again that the "oracle1" directory ends with a one, not an "ell".

```
# cd /etc/cmcluster
# mkdir oracle1
# cd oracle1
```

2. **[Node1/root]** Next, create the package template file for the oracle package, named oracle1.

```
# cmmakepkg -v -p oracle1.conf
```

3. **[Node1/root]** Edit this package configuration file as described below.

```
# vi oracle1.conf
```

and modify the following parameters:

PACKAGE_NAME	oracle1
NODE_NAME	<node1>
NODE_NAME	<node2>
RUN_SCRIPT	/etc/cmcluster/oracle1/oracle1.cntl
RUN_SCRIPT_TIMEOUT	NO_TIMEOUT
HALT_SCRIPT	/etc/cmcluster/oracle1/oracle1.cntl
HALT_SCRIPT_TIMEOUT	NO_TIMEOUT
SERVICE_NAME	ORACLE1
SERVICE_FAIL_FAST_ENABLED	NO
SERVICE_HALT_TIMEOUT	300
SUBNET	10.x.x.x ← Use netstat -in to get the 10-net subnet address

When finished, save your changes and exit the file.

4. [Node1/root] Create the **run/halt** script template.

```
# cmmakepkg -v -s oracle1.cntl
```

5. [Node1/root] Edit the package control script template as described below.

```
# vi oracle1.cntl
```

and modify the following parameters:

```
VG[0] = "vg03"

LV[0] = "/dev/vg03/oracle1"
FS[0] = "/oracle1"
FS_MOUNT_OPT[0] = ""
FS_UNMOUNT_OPT[0] = ""
FS_FSCK_OPT[0] = ""
FS_TYPE[0] = "vxfs"

IP[0] = "10.x.x.x"           → get a 10-net IP address from your instructor
SUBNET = "10.y.y.y"          → use netstat -in to obtain the 10-net subnet

SERVICE_NAME[0] = "ORACLE1"
SERVICE_CMD[0] = "/etc/cmcluster/oracle1/haoracle.sh monitor"
SERVICE_RESTART[0] = "-r 0"
```

```
function customer_defined_run_cmds {
# ADD customer defined run commands

    /etc/cmcluster/oracle1/haoracle.sh start
    test_return 51
}
```

```
function customer_defined_halt_cmds {
# ADD customer defined run commands

    /etc/cmcluster/oracle1/haoracle.sh shutdown
    test_return 52
}
```

When finished, save your changes and exit the file.

## Part 5: Create the Oracle Monitoring Script

1. [Node1/root] Place a copy of the default monitoring script in the oracle1 directory.

```
# cd /etc/cmcluster/oracle1  
# cp /opt/cmcluster/toolkit/oracle/haoracle* .
```

2. [Node1/root] Update the haoracle.conf file using the values in the table below:

ORACLE_HOME=/opt/oracle/product/10.1.0.db
SID_NAME=ORA1SID
LISTENER_NAME=LISTENERORA1

Save your changes and exit the file.

## Part 6: Copy the Control and Monitor Scripts to Second System

1. [Node1/root] On your primary system, ensure that the haoracle.sh and oracle1.cntl scripts are both executable.

```
# cd /etc/cmcluster/oracle1  
# chmod 755 oracle1.cntl  
# chmod 755 haoracle.sh
```

2. [Node2/root] On <node2>, create the oracle package directory and copy the appropriate files from <node1>.

```
# cd /etc/cmcluster  
# mkdir oracle1  
# cd oracle1  
# rcp <node1>:/etc/cmcluster/oracle1/* .
```

## Part 7: Finish the Configuration of the Oracle Package

1. [Node1/root] Back on <node1>, modify the cluster-wide ASCII configuration file to include our new **vg03** volume group.

```
# cd /etc/cmcluster  
# vi cmclconfig.ascii
```

← add vg03 at the end of the file

2. [Node1/root] Check both the cluster-wide configuration file and the Oracle package configuration file for errors.

```
# cd oracle1  
# cmcheckconf -v -P oracle1.conf -C ./cmclconfig.ascii
```

3. [Node1/root] Fix any errors. Once configuration is OK, apply the new configuration file.

```
# cmapplyconf -v -P oracle1.conf -C ./cmclconfig.ascii
```

4. [Node1/root] Once the new configuration is applied, start the **oracle1** package.

```
# vgchange -a n vg03      ← Be sure vg03 is not activated on either node  
# cmmodpkg -v -e oracle1
```

**The Highly Available Oracle Database**

5. [Node1/oracle] On whichever node the oracle1 package is running (it should be <node1>), check the database status and contents.

```
# su - oracle
$ /labs/mod_13/test_oracle.sh
```

You should see output that looks like the following:

```
$ /labs/mod_13/test_oracle.sh

SQL*Plus: Release 10.1.0.2.0 - Production on Wed Jun 8 08:15:26
2005

Copyright (c) 1982, 2004, Oracle. All rights reserved.

Connected to:
Oracle Database 10g Enterprise Edition Release 10.1.0.2.0 - 64bit
Production
With the Partitioning, Oracle Label Security, OLAP and Data Mining
Scoring Engine options

SQL>
      NUM NAME          CITY
----- -----
      0 MJ            Chicago; IL
      1 Magic         Los Angeles, CA
      2 Bird          Boston, MA
      3 Gary          Detroit, MI
      4 Martin        Frankfurt, Germany
      5 Vince         Toronto, Canada

6 rows selected.

SQL> Disconnected from Oracle Database 10g Enterprise Edition
Release 10.1.0.2.0 - 64bit Production
With the Partitioning, Oracle Label Security, OLAP and Data Mining
Scoring Engine options

$
```

**Be sure to exit from the oracle login before continuing**

```
$ exit
```

## Part 8: Manually Test Serviceguard Failover of the Oracle Package

1. [Node1/root] First, verify that the oracle1 package can start on the alternate node correctly.

```
# cmhaltpkg -v oracle1
# cmrumpkg -v -n <node2> oracle1
```

2. [Node2/oracle] Now, on <node2> (where the oracle1 package should be running, check the database status and contents.

```
# su - oracle
$ /labs/mod_13/test_oracle.sh
```

As before, you should see output that looks like the following:

```
$ /labs/mod_13/test_oracle.sh

SQL*Plus: Release 10.1.0.2.0 - Production on Wed Jun 8 08:15:26
2005

Copyright (c) 1982, 2004, Oracle. All rights reserved.

Connected to:
Oracle Database 10g Enterprise Edition Release 10.1.0.2.0 - 64bit
Production
With the Partitioning, Oracle Label Security, OLAP and Data Mining
Scoring Engine options

SQL>
      NUM NAME          CITY
      ----- -----
      0 MJ            Chicago; IL
      1 Magic         Los Angeles, CA
      2 Bird          Boston, MA
      3 Gary          Detroit, MI
      4 Martin        Frankfurt, Germany
      5 Vince         Toronto, Canada

6 rows selected.

SQL> Disconnected from Oracle Database 10g Enterprise Edition
Release 10.1.0.2.0 - 64bit Production
With the Partitioning, Oracle Label Security, OLAP and Data Mining
Scoring Engine options

$
```

**Be sure to exit from the oracle login before continuing**

```
$ exit
```

3. [Node1/root] Verify the oracle1 package as it starts up on the alternate node in the cluster. Also, re-enable AUTO\_RUN switch for your oracle1 package.

```
# cmviewcl -v -p oracle1  
# cmmodpkg -v -e oracle1
```

If the package is not running correctly on either node, troubleshoot the problem.

4. Congratulations on successfully configuring your first Oracle package!

## 13-9. LAB: Test Oracle Package Failover

- From the node on which your `oracle1` package is currently running, and as the `oracle` user, execute the `loop_oracle.sh` script, which will write to the database once a second.

```
# cmviewcl -vp oracle1           ← To determine on which node oracle1 is running
# su - oracle
$ /labs/mod_13/loop_oracle.sh
```

Allow the `loop_oracle.sh` script to continue running as you view the script contents. Then, while the `loop_oracle.sh` script is still running, continue to step 2 in this lab.

The `loop_oracle.sh` script contains the following code:

```
#####
#
# This script inserts rows into table test.
#
i=10
while true
do
    sqlplus -s system/manager <<-! | grep 'ORA-01034'
    set echo on
    set termout on
    whenever sqlerror exit 5;
    insert into test
        values ($i, $i || ' Any Name', $i || ' Any City');
    commit;
    exit;
!
if [[ $? -eq 0 ]]
then
    echo "Oracle has exited"
    exit
else
    echo "$i occurrence in loop"
    i=$(expr $i + 1)
    sleep 2
fi
if [[ $i -gt 200 ]]
then
    exit
fi
done
```

**The Highly Available Oracle Database**

2. Allow the `loop_oracle.sh` script to continue to run. Now, from the node on which your oracle1 package is currently running, open a new terminal window and prepare to kill the `haoracle.sh` service process. In the window in which the `loop_oracle.sh` script is running, note the last line number written to the database at the moment you execute the following `kill` command. Write that number below.

```
# ps -ef | grep haoracle.sh  
# kill <pid>
```

Enter here the line number of the last line written to the database → \_\_\_\_\_

3. Verify that the `oracle1` package fails over to the other node in the cluster.

```
# cmviewcl -vp oracle1
```

4. From the node where the `oracle1` package is **now** running, verify that data in the database table experienced no data loss as follows: The last number written to the database (as seen in the output of the following script) should match the last number, as logged in step 2 above, thus proving that there was no loss of data during the failover. When you are finished viewing the Oracle data, exit the `oracle` account.

```
# su - oracle  
$ /labs/mod_13/test_oracle.sh  
$ exit
```

5. Finally, to allow for future failovers, re-enable your node switching parameter for the node where the `oracle1` package was previously running.

```
# cmmodpkg -v -e -n <nodeX> oracle1
```

---

## **Module 14 — EMS Resources and Serviceguard Packages**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- List the five EMS monitoring daemons.
- Use the `resls` command to list the available EMS resources.
- Set up a Serviceguard package to use EMS resources.

## 14-1. SLIDE: Review of Client/Server Management Architecture

### Review of Client / Server Management Architecture

The diagram illustrates the SNMP client/server architecture. On the left, a server is represented by a rack-mounted hardware unit. A dotted arrow points from the server to a central horizontal line, labeled "SNMP-get Packet". This packet is sent to a client on the right, which is shown as a computer monitor and keyboard. The client also has a dotted arrow pointing back to the same line, labeled "SNMP-response Packet". To the right of the client, a box labeled "MIB" represents the Management Information Base. The HP Invent logo is in the top right corner of the slide.

Types of SNMP Packets	
• SNMP-get	( server initiated )
• SNMP-set	( server initiated )
• SNMP-response	( client response )
• SNMP-trap	( client initiated )

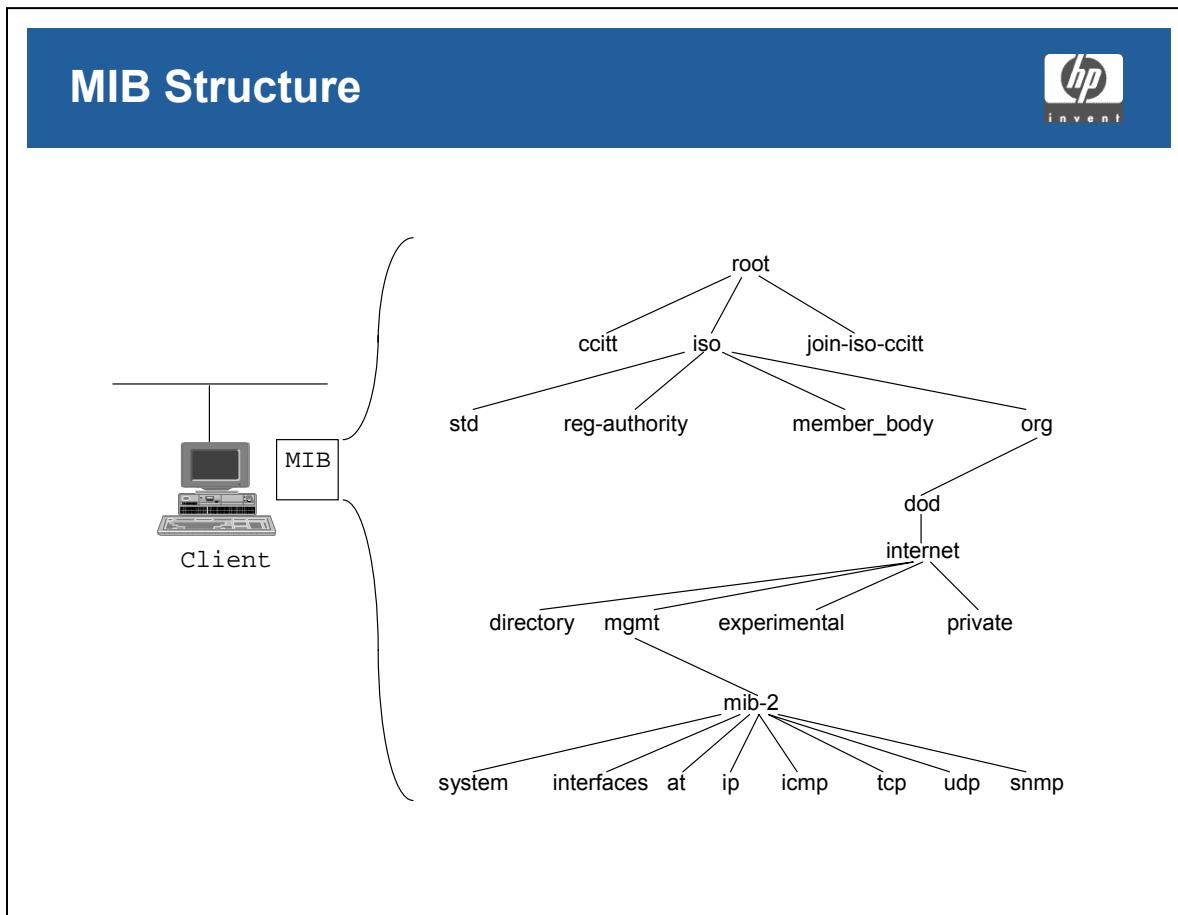
### Student Notes

A client/server monitoring protocol, which has been around for a long time, is the SNMP design. This is the design in which EMS (Event Monitoring Service) monitoring is based upon.

The SNMP protocol was designed to allow one system (a server) to monitor many other systems (clients) remotely, easily, and with as little overhead as possible for the server.

With the SNMP protocol, every client maintains a data structure called a Management Information Base (MIB), which contains many variables related to the state and status of the client. If a server wants to retrieve a specific MIB variable from a client, the server initiates an SNMP-get packet to the client. The client then returns the MIB value through an SNMP-response packet.

## 14-2. SLIDE: MIB Structure

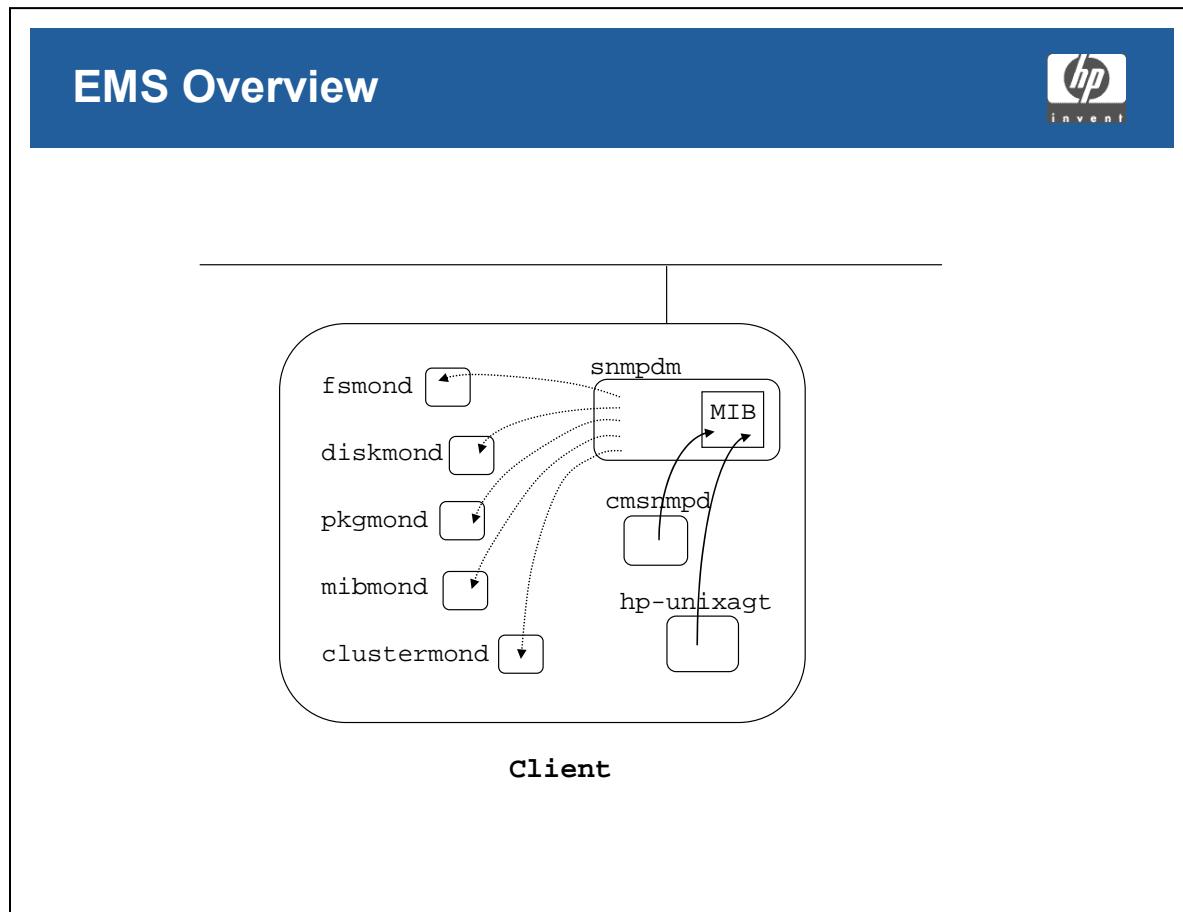


### Student Notes

The slide contains a general, high-level diagram of the MIB data tree structure maintained on each client system. There are hundreds of different variables maintained in this structure, including system information, NIC status information, and statistics related to the IP, TCP, UDP, and ICMP protocols.

The MIB structure is maintained in the data area of the `snmpd` process (also known as the SNMP master daemon).

### 14-3. SLIDE: EMS Overview



### Student Notes

The EMS product provides local monitoring services for applications running on the local system.

The EMS product works by starting five daemons, which monitor variables in the local MIB, and are kept and maintained by the SNMP master daemon.

The five EMS daemons are:

- **fsmond** Monitors file system variables like percent full and MB available.
- **diskmond** Monitors disk variables like driver status and disk errors.
- **mibmond** Monitors general network variables like NIC status and packet counts.
- **pkgmond** Monitors Serviceguard package variables.
- **clustermond** Monitors Serviceguard cluster variables.

## **14-4. TEXT PAGE: EMS Resource Class Hierarchy**

Online documentation about EMS is available in the following entries:

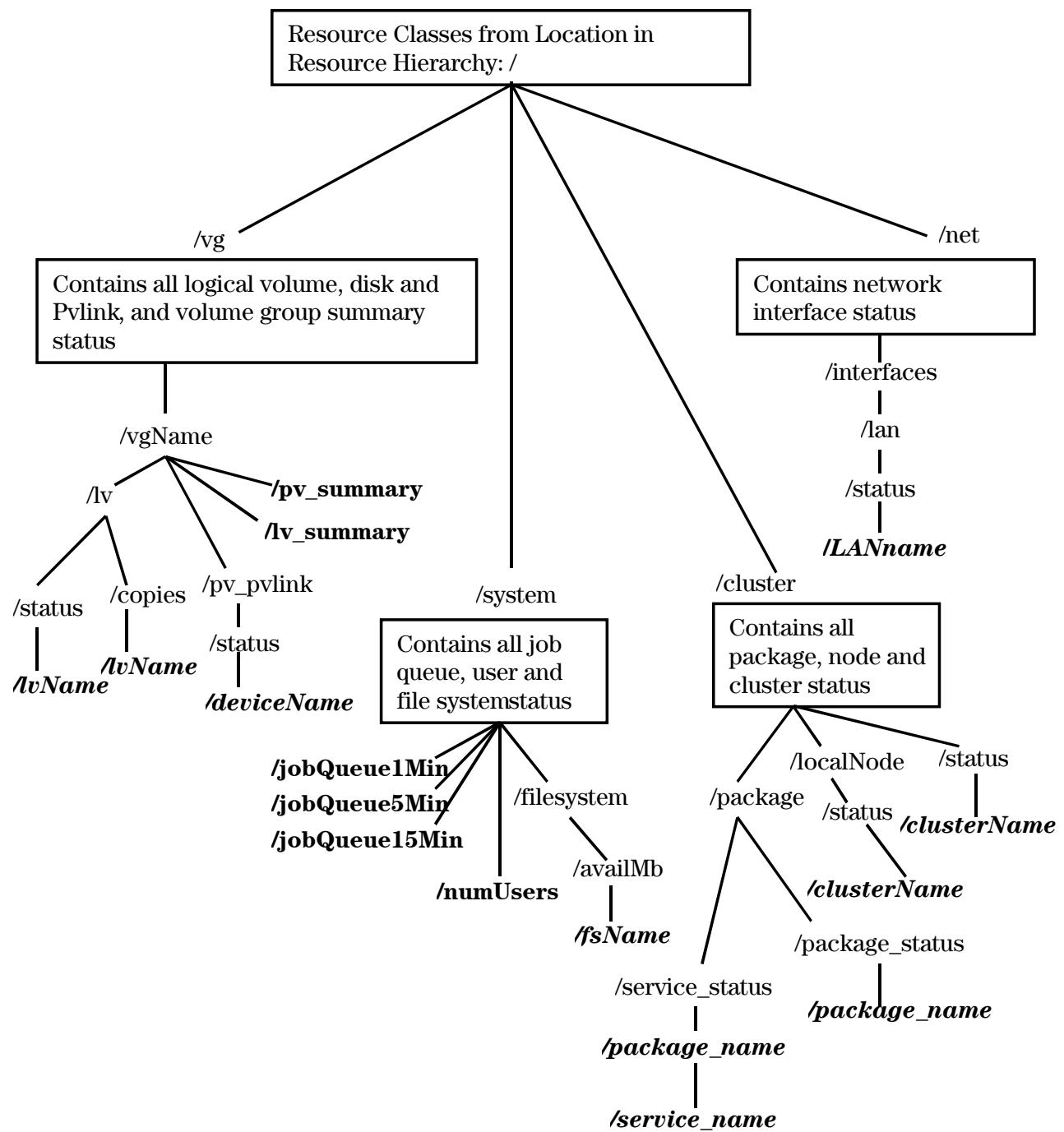
`man 5 ems`, `man 1 resls`, `man 1m diskmond`, and `man 1m mibmond`

The full path of a resource includes the resource class hierarchy and instance. The figure shows an example of a full resource path for the physical volume status of the device

`/dev/dsk/c0t1d2` belonging to volume group `vgDataBase` is  
`/vg/vgDataBase/pv_pvlink/status/c0t1d2`

The `resls` command takes a resource pathname as an argument and displays elements under that pathname that are being monitored. “/” is the top of the resource path tree.

The following chart shows the resource hierarchy.



**Bold Italics** - Instances, replaced with an actual name  
**Bold** - Resource instance

## 14-5. SLIDE: Setting Up a Package to Use an EMS Resource

### Setting Up a Package to Use an EMS Resource



1. Use the `resls` command to identify the EMS resource name.
2. Edit the package configuration file and add the resource definition.

```
vi /etc/cmcluster/package/pkg.conf
```

```
RESOURCE_NAME          /system/filesystem/availMb/home
RESOURCE_POLLING_INTERVAL 30
RESOURCE_START        AUTOMATIC
RESOURCE_UP_VALUE     >= 5
```

3. Halt the package.
4. Reapply the package configuration file.
5. Restart the package.

### Student Notes

The above slide shows the procedure to setup a package to use an EMS resource.

The `resls` command can be used to list the different resource variables.

To list an EMS resource, the full name of the resource must be specified. For example, to list the available megabytes currently free in the `/home` file system, the command is:

```
# resls /system/filesystem/availMb
```

---

**NOTE:** The default `RESOURCE_POLLING_INTERVAL` is 60 seconds. (The minimum is 30 seconds).

---

A feature first introduced with Serviceguard version 10.10 is the ability to make a package dependent upon an EMS resource (a MIB variable).

## EMS Resources and Serviceguard Packages

In order to do this, the `cmcld` daemon needs to know if any packages are defined containing EMS resources. The package variables are:

```
RESOURCE_NAME  
RESOURCE_POLLING_INTERVAL  
RESOURCE_START  
RESOURCE_UP_VALUE
```

If a package does have an EMS resource defined, the `cmcld` must register with the appropriate EMS monitoring daemons during its initialization. By registering, the EMS daemons return the current value of the variable every `RESOURCE_POLLING_INTERVAL` seconds.

If the `RESOURCE_NAME` variable ever returns with a value outside the range of `RESOURCE_UP_VALUE`, then the `cmcld` daemon will failover the package to a node where the `RESOURCE_UP_VALUE` is within specified parameters.

The `RESOURCE_POLLING_INTERVAL` indicates how often, in seconds, the resource is to be monitored. The default `RESOURCE_POLLING_INTERVAL` is 60 seconds if not specified. The minimum `RESOURCE_POLLING_INTERVAL` is 30 seconds.

The `RESOURCE_START` value may be set to either `AUTOMATIC` or `DEFERRED`. The default setting is `AUTOMATIC`. A setting of `AUTOMATIC` will mean that Serviceguard will automatically start up resource monitoring when the node starts up. If `DEFERRED` is selected, Serviceguard will not attempt to start resource monitoring during node start up. In this case, the Serviceguard administrator should specify all `DEFERRED` resources in the package run script so that these `DEFERRED` resources will be started up from the package run script when the package starts up.

The `RESOURCE_UP_VALUE` requires an operator and a value. The combination of these two items defines the resource "UP" condition. The operators are `=`, `!=`, `>`, `<`, `>=`, and `<=`, depending on the type of value. These values can be string or numeric. If the type is string, then only `=` and `!=` are valid operators. If the string contains whitespace, it must be enclosed in quotes. The following are some examples of the correct use of the `RESOURCE_UP_VALUE` ...

RESOURCE_UP_VALUE	= UP
RESOURCE_UP_VALUE	<code>!=</code> DOWN
RESOURCE_UP_VALUE	= 5
RESOURCE_UP_VALUE	> 5.1
RESOURCE_UP_VALUE	> -5 and < 10

---

## 14–6. LAB: Monitoring File Systems with EMS

### Directions

This lab uses EMS (Event Monitoring Service) to monitor the free disk space available in a new file system. This Resource Monitor will be added to the existing package logprog.

1. First, ensure you have all EMS products loaded on all nodes in your cluster. Check and verify that you have the following five products already installed:

- EMS-Config,
- EMS-Core,
- EMS-DBMon,
- EMS-DskMon, and
- EMS-MIBMon.

#### Answer:

```
# swlist -l product | grep -i ems
```

2. If your systems are missing any of the above products, `swinstall` them now.

#### Answer:

```
# swinstall
```

3. Perform the following steps on both nodes in your cluster.

In volume group `vg00`, create a new 64MB logical named `ems_test`. Build a `vxfs` file system in the `ems_test` logical volume. Make a `/ems_test` mount point directory and mount your new file system.

#### Answer:

```
<node1># lvcreate -L 64 -n ems_test vg00
<node1># newfs -F vxfs /dev/vg00/remstest
<node1># mkdir /ems_test
<node1># mount /dev/vg00/ems_test /ems_test

<node2># lvcreate -L 64 -n ems_test vg00
<node2># newfs -F vxfs /dev/vg00/remstest
<node2># mkdir /ems_test
<node2># mount /dev/vg00/ems_test /ems_test
```

Module 14  
**EMS Resources and Serviceguard Packages**

4. Next, halt the logprog package.

**Answer:**

```
# cmhaltpkg -v logprog
```

5. Edit the logprog.conf file to add a resource to the logprog package. Afterwards, check and apply your logprog.conf file

```
# cd /etc/cmcluster/logprog
# vi logprog.conf
```

RESOURCE_NAME	/system/filesystem/availMb/ems_test
RESOURCE_POLLING_INTERVAL	30
RESOURCE_START	AUTOMATIC
RESOURCE_UP_VALUE	>= 48

```
# cmcheckconf -v -P logprog.conf
# cmaplyconf -v -P logprog.conf
```

6. To test our new EMS resource, bring up the logprog package.

**Answer:**

```
# cmmodpkg -v -e logprog
```

7. Let us now confirm the resource monitoring.

**Answer:**

```
# cmviewcl -v -p logprog
```

8. To test the EMS failover capability, copy a fairly large file (eg. /stand/vmunix) into the /ems\_test file system. This file system is 64 MB in size. An HP-UX 11.23 kernel file is about 53 MB in size. An HP-UX 11.00 kernel file is about 25 MB in size. Either way, after copying, the available free space in the /ems\_test file system should be well under the 48 MB limit, as specified by the RESOURCE\_UP\_VALUE. This will then cause the logprog package to failover.

**Answer:**

```
# cp /stand/vmunix /ems_test/.
```

9. After the failover, execute a `cmviewcl` on the `logprog` package. Note the condition of the `logprog` package node switches.

### **Answer:**

```
# cmviewcl -v -p logprog
```

10. As you have time and interest, continue to experiment with the EMS resources. For example, what happens to the logprog package if the RESOURCE\_UP\_VALUE is violated on both nodes simultaneously?

## Answer:

Run various experiments.

11. To return the cluster to its previous state, remove the resource from your `logprog` package. Also, remove all copies of `vmunix` from the `/ems_test` file system(s).

## Answer:

Re-comment out the following lines in logprog.conf:

```
#RESOURCE_NAME  
#RESOURCE_POLLING_INTERVAL  
#RESOURCE_START  
#RESOURCE_UP_VALUE
```

Then: # cmcheckconf -P /etc/cmcluster/logprog/logprog.conf  
# cmapplyconf -P /etc/cmcluster/logprog/logprog.conf

## **14-7. LAB: Monitoring *numUsers* with EMS**

### **Directions**

#### **Special Note**

- a. This lab demonstrates the use of EMS and Serviceguard to monitor "numUsers."
- b. **VERY IMPORTANT:** If you are using remote equipment, this lab will perform correctly only when using an `hpterm` to log into Unix. The lab will not work correctly using "`XDMCP DIRECT or Putty`". To simulate multiple logins, use ReflectionX.

1. First, halt the `logprog` package.

**Answer:**

```
# cmhaltpkg -v logprog
```

2. Modify the `logprog.conf` file as follows:

```
# cd /etc/cmcluster/logprog
# vi logprog.conf
```

Set the following values:

RESOURCE_NAME	/system/numUsers
RESOURCE_POLLING_INTERVAL	60
RESOURCE_START	AUTOMATIC
RESOURCE_UP_VALUE	< 4

Note that the above lines must be *uncommented*, and they must have values added to the end of each line.

Also note that, if you have several windows open already, the `RESOURCE_UP_VALUE` may need to be modified upward. Choose a value that is 2 or 3 greater than the number of windows currently open.

3. Now, check the `logprog.conf` file for errors, apply it, and bring up the `logprog` package.

**Answer:**

```
# cmcheckconf -v -P logprog.conf
# cmapplyconf -v -P logprog.conf
# cmmodpkg -v -e logprog
```

4. To test this new resource, do several logons to the node running the `logprog` package. Create enough separate logins (or, open enough additional `hpterm` windows) to violate the `RESOURCE_UP_VALUE`. This should cause the package to move over to the adoptive node. You should also notice the state of this resource in `cmviewcl -v`.

**Answer:**

Create enough separate logins to violate the `RESOURCE_UP_VALUE`. Remember the “Special Note” at the beginning of this lab concerning the use of `hpterm`s only.  
Afterward, ...

```
# cmviewcl -v -p logprog
```

5. To return the cluster to its previous state, remove the four “RESOURCE” lines from the `logprog` package configuration file. Then `cmcheckconf` and `cmapplyconf` your `logprog` package. Finally, start your `logprog` package.

**Answer:**

```
# vi logprog.conf
# cmcheckconf -v -P logprog.conf
# cmapplyconf -v -P logprog.conf
# cmmodpkg -v -e logprog
```

**Some Final Notes**

- a. Package switches are unaffected, even after the `logprog` package fails over to an adoptive node due to resource failure.
- b. No logging information is found in the package logs; all logging is in `syslog.log`.
- c. A package cannot even start on a node in which the resource is not met.



---

# **Module 15 — High Availability Networking**

## **Objectives**

Upon completion of this module, you will be able to do the following:

- Connect a second network to a Serviceguard cluster with multiple levels of redundancy.
- Set up a Serviceguard node to provide LAN card redundancy.
- Configure a highly available network using redundant hubs.
- Configure a highly available network using redundant routers.
- Configure a highly available network using redundant networks.

## 15-1. SLIDE: Network Redundancy

### Network Redundancy



- Redundant LAN cards
- Redundant hubs
- Redundant routers
- Redundant networks

### Student Notes

In order to provide a high level of availability, a typical cluster should remain accessible to client systems even after a failure of any network component.

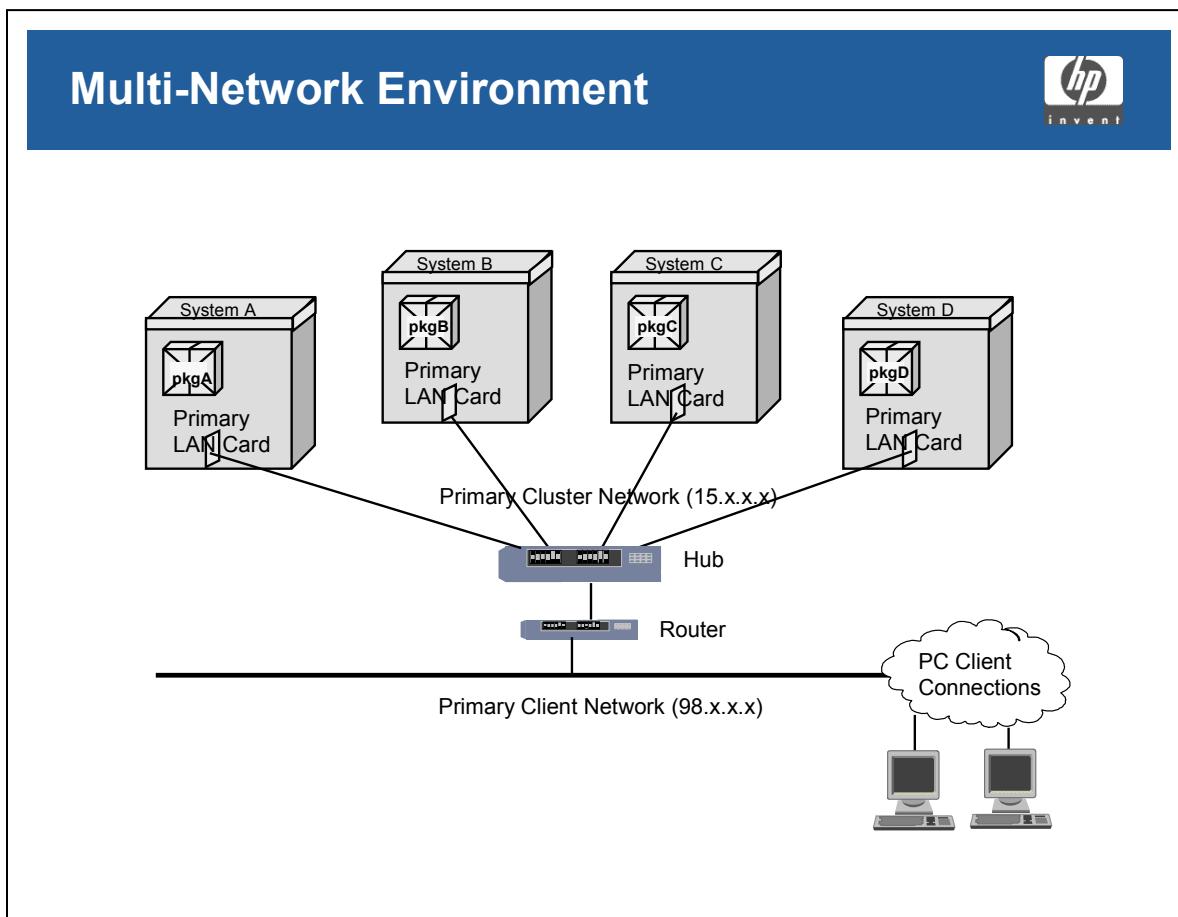
The key to eliminating SPOF ( within a network ) is understanding the topologies and the failure points within those topologies.

For most customers, it is not cost effective to eliminate every single possible point of failure within their network for every single node. In a Serviceguard environment, the nodes within the cluster ( at the very least ) should be made highly available from a networking standpoint. Any LAN card, LAN segment, hub, or router failure, should not cause communication among nodes within the cluster to fail.

For other nodes within the network, it will depend on the type of node and how that node is used as to whether the node should be highly available.

This module covers how SPOF in the network can be eliminated from the server side and how to segment the clients such that a network failure only affects a small portion of the users.

## 15-2. SLIDE: Multi-Network Environment



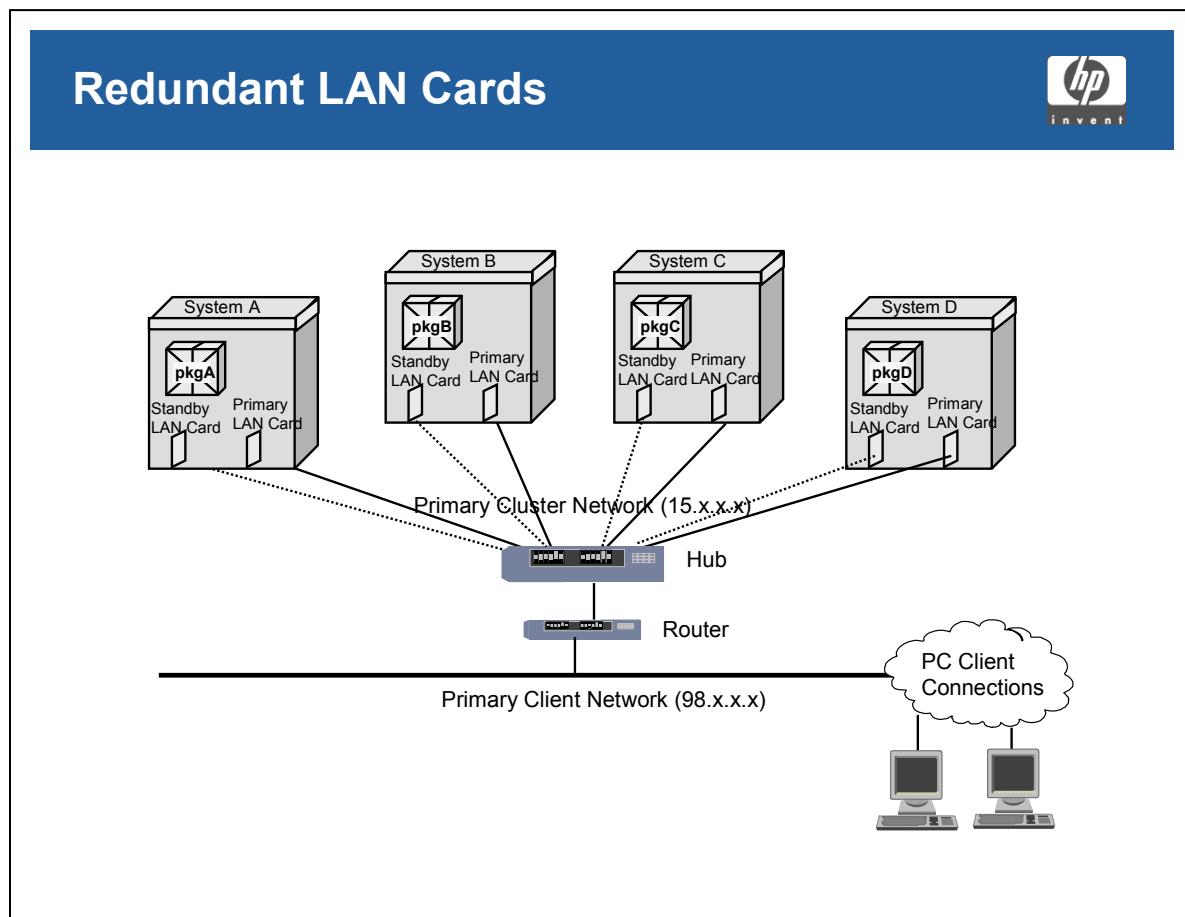
### Student Notes

Typically, a Serviceguard network is not on the same network as the client systems. In many cases, a firewall separates clients from the cluster to prevent unauthorized access to the cluster.

The next few slides look at different issues/concerns when connecting a second network ( i.e. a client network ) to a Serviceguard network.

In the above slide, identify four potential SPOF that exist in the network connection between the Serviceguard servers and the PC Client systems.

### 15–3. SLIDE: Redundant LAN Cards



### Student Notes

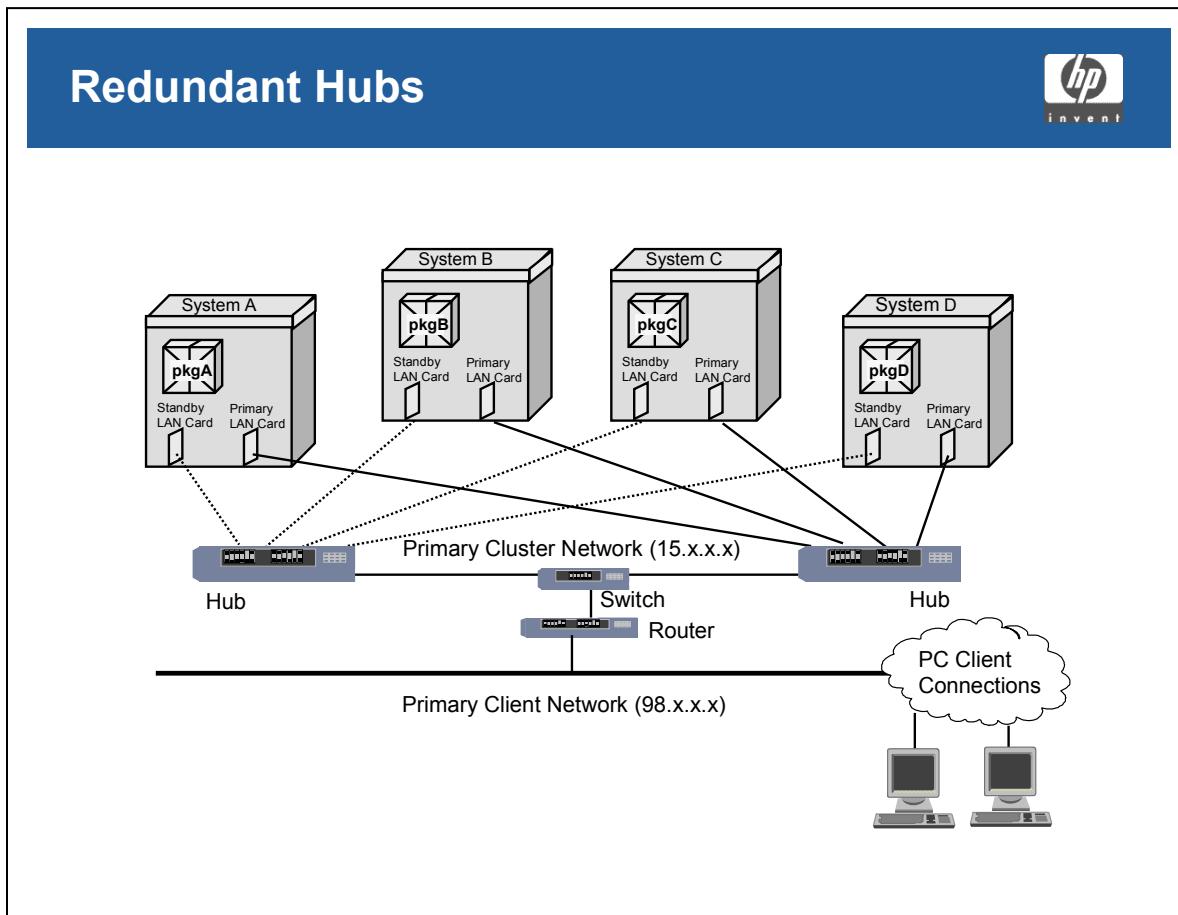
The first step towards eliminating single points of failure for networking is to ensure each node in the cluster has redundant network interface cards. A second network interface card also protects against a cable failure between the node and the hub.

The IP addresses for the packages and the node are assigned to the primary network interface card. The second interface card initially does not have any IP addresses assigned to it, and it is referred to as the "standby" interface card.

Serviceguard uses the standby card if it detects a failure with network communications on the primary card. The network communications failure could be a failed interface card, or a bad cable, or a down device connecting the node to the client network, like a hub or router.

Whenever Serviceguard detects no communications is possible through the primary interface card, it switches all IP addresses over to the standby interface card.

## 15-4. SLIDE: Redundant Hubs



### Student Notes

If the cluster is cabled using hubs (as shown on the slide), a minimum of two hubs should be used to avoid a SPOF in the hub.

Since a router is not allowed to have two connections to the same network, the router in the above slide is not allowed to have connections to two hubs. Hence, a switch is used to connect both hubs to the router.

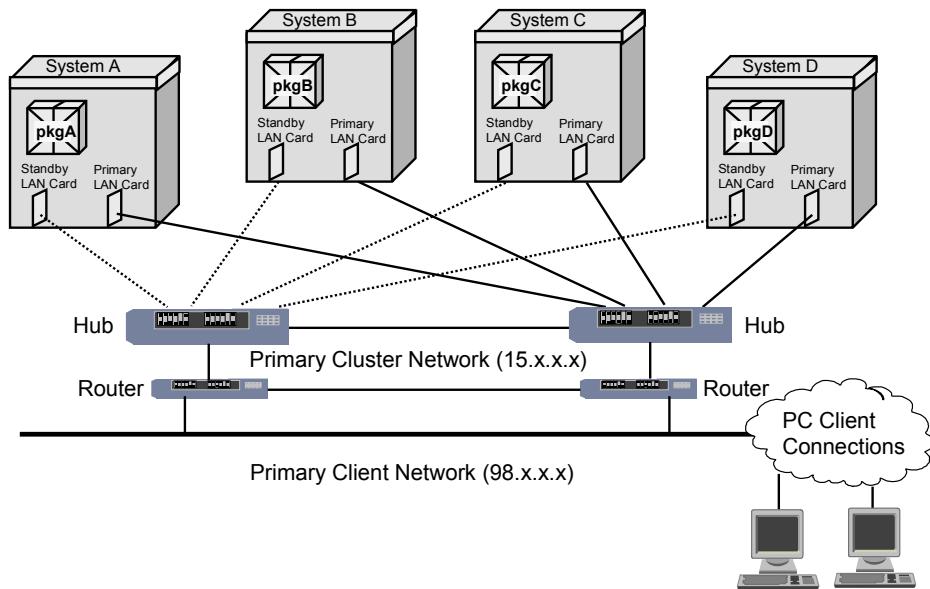
In the slide, communications among cluster nodes will survive any networking related failure, including LAN cards, twisted pair cables, hubs, or network backbone.

**NOTE:** If the network backbone breaks, communication among the cluster nodes continues because the cluster nodes are attached to the same hubs. However, from the perspective of clients connecting to the cluster via the backbone, the backbone failure would cause all communications with the cluster to cease.

In this situation, a second LAN segment should be added, as shown in an upcoming slide.

## 15–5. SLIDE: Redundant Routers

### Redundant Routers

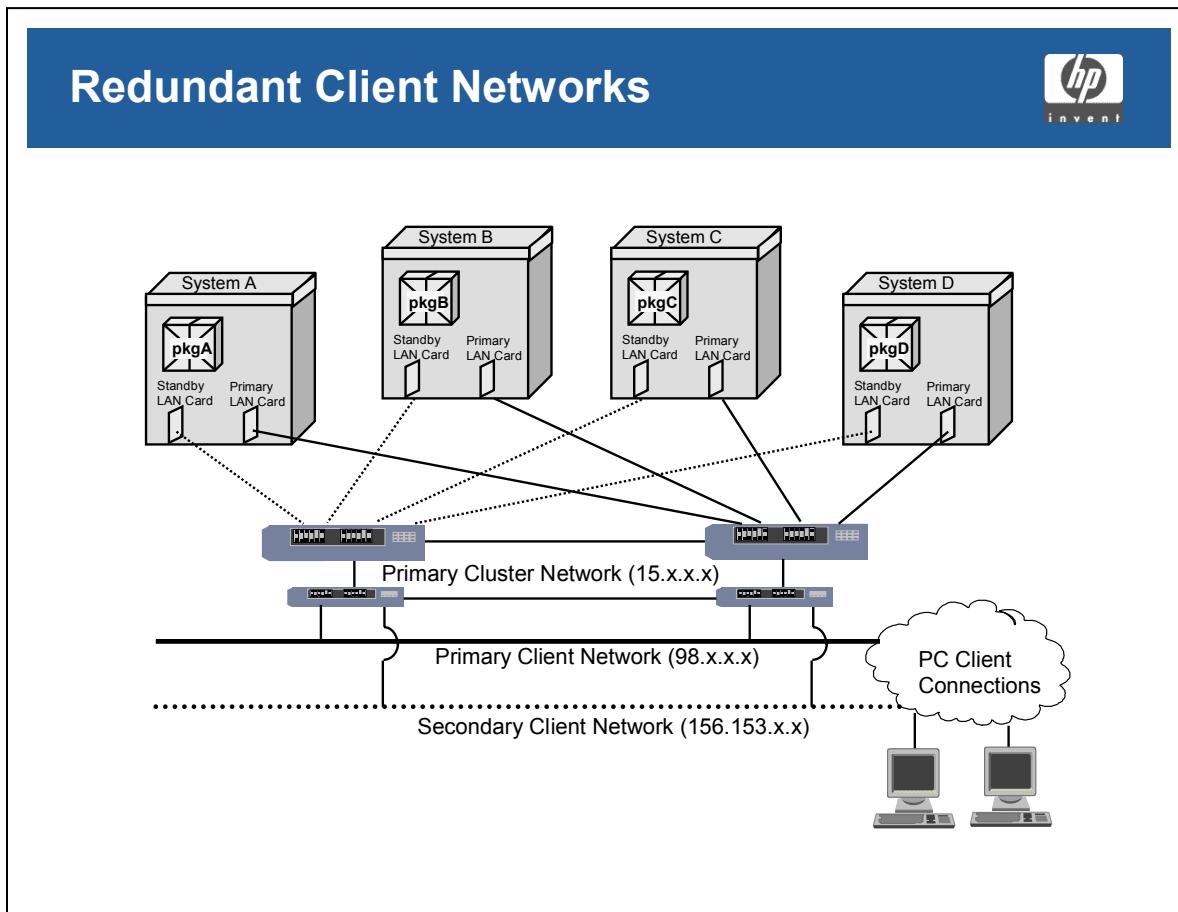


### Student Notes

If a single router is used, the client / cluster communications will have a SPOF at the router. To eliminate the SPOF, a second router between the clients and the cluster is added as shown on the slide.

Each client has its own route table entry to get to the cluster. Some clients may have entries for the first router; other clients could have entries for second router. Should a router fail, routing information protocol ( RIP ) packets would inform any clients using the failed router to now use the other router for their cluster communications.

## 15-6. SLIDE: Redundant Client Networks



### Student Notes

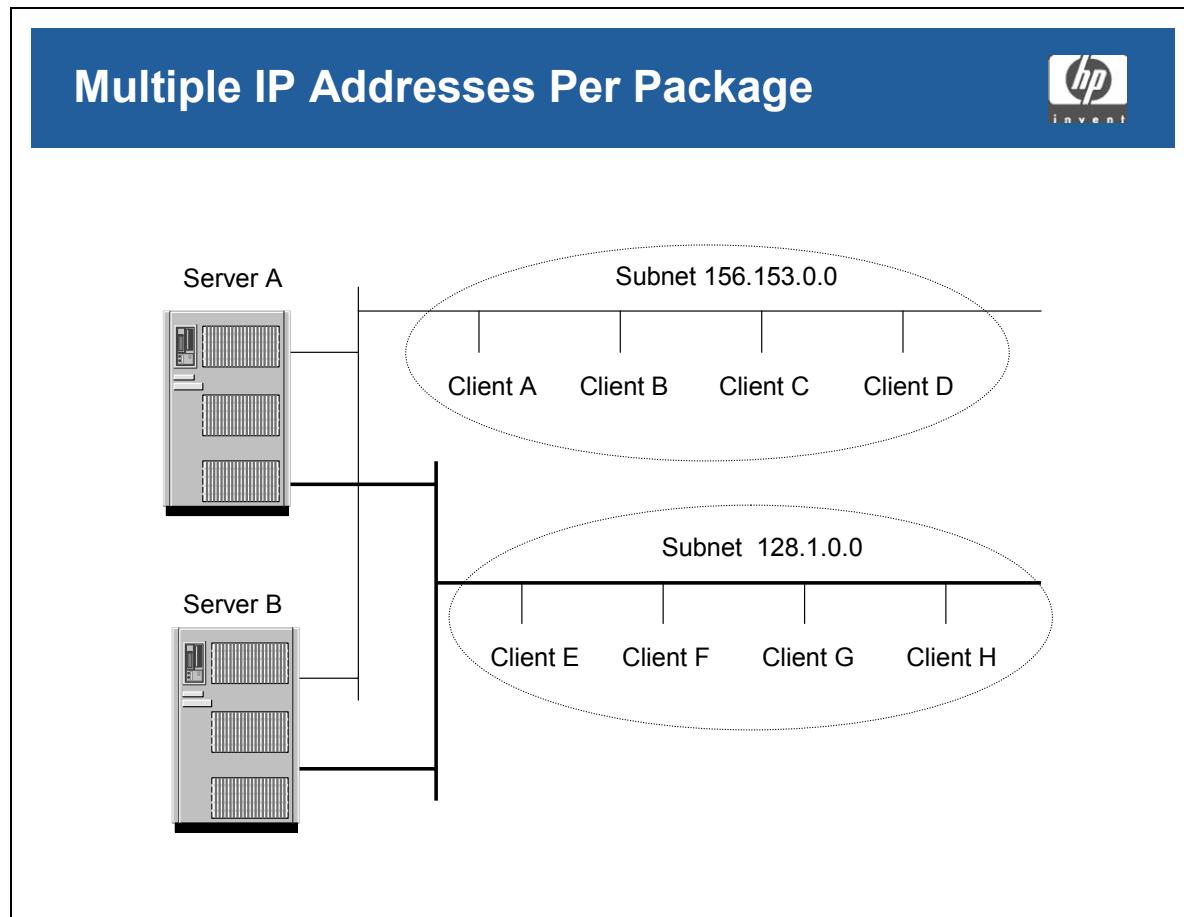
The final component that needs to be made highly available is the client network itself. This is accomplished by connecting both routers to a second network that belongs to the group of network associated with client networks.

By having multiple client networks being able to connect to the Serviceguard clustered network, the communications to the cluster no longer has a single-point-of-failure (SPOF) with the client network itself.

The above slide shows a network configuration where the four main SPOFs related to client-cluster communications have been eliminated. The four SPOFs which have been eliminated are:

- LAN Card failures
- Hub failures
- Router failures
- Client network failures

## 15-7. SLIDE: Multiple IP Addresses



### Student Notes

A package can have multiple floating IP addresses assigned to it (on different subnets) to better balance traffic across multiple subnets. The package would have one IP address for each subnet on which clients reside. Each client would point to the floating IP address on its own subnet.

## 15-8. LAB: Losing Heartbeat Packets

### Directions

1. With the cluster up, both nodes as members, and  
`tail -f /var/adm/syslog/syslog.log` running on both servers ...
  - a. **If you are using remote equipment**, cd into the `/labs/mod_15` and then run the `"simulate_lan_card_failure3.sh"` script **on one node** of your cluster. This is similar to the script that you have executed in Module 6. Here, however, this script will disable all LAN cards on one node.
  - b. **If you are using local equipment**, then disconnect **all** LAN card cables on one node from the network.

Note that both systems are up, Serviceguard daemons are running on both nodes, but neither node is receiving heartbeats (both nodes think the other node has failed). In this situation, if the cluster splits into two, only one subcluster will continue and the other subcluster will **TOC** (panic). If one set of nodes consists of more than half the cluster, these nodes adopt all applications configured for them. All other nodes (with less than half) **TOC** to protect against the same applications running on two different systems.

If the pieces are exactly 50% of the cluster (as is the case here) the lock disk is needed to break the tie. Both servers will attempt to access the lock disk. One server obtains the lock disk and continues to run the cluster as a single node cluster, the other node should **TOC**.

**Answer:**

```
# cd /labs/mod_15
# ./simulate_lan_card_failure3.sh -or- disconnect ALL LAN cables on one node.
```

2. After one of the nodes TOCs, ...
  - a. **If you are using remote equipment**, hit <CNTL-C> (if needed) to exit out of the `"simulate_lan_card_failure3.sh"` script on the surviving node.
  - b. **If you are using local equipment**, then reconnect the LAN cables on the same node.

**Answer:**

<CNTL-C> --or-- reconnect LAN cables on ServerA.

Module 15  
**High Availability Networking**

---

## **Module 16 — Rolling Upgrade Issues**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- Explain why rolling upgrades are carried out.
- List provisions for rolling upgrades.
- Know when rolling upgrades will fail.
- Carry out a rolling upgrade.

## 16-1. SLIDE: Minimizing Planned Downtime

### Minimizing Planned Downtime



- Faster system upgrades / patches
- Faster application updates / patches
- On-line application reconfiguration
- More accurate package configuration documentation
- More detailed package sign-off and testing

### Student Notes

Planned downtime is scheduled. As a system administrator you strive to keep this time to an absolute minimum. The most common reasons for planned downtime are for disk management, operating system upgrades, package updates and hardware upgrades. All of these points can be addressed by various means. File system management can be done using OnLine JFS. Operating system, hardware and package updates/upgrades/maintenance can be done using rolling upgrades.

### Reducing Time for Operating System/Application Upgrades

A rolling upgrade is the method used to minimize the amount of planned downtime. Rolling upgrades can take many forms. In most cases it involves transferring all packages from the node which requires maintenance to another node in the cluster. For example, in a client server model, the database server could be upgraded on one day, causing one hour of downtime. Other application servers could be upgraded one at a time throughout the week, maintaining the highly available environment.

In your initial cluster specification, you must allow for large outages caused when packages from a failed node add extra load to their adoptive nodes. Having a "spare node" in the cluster allows you to eliminate this additional load when you have to remove a node from the

cluster for upgrade. The spare node is configured as an adoptive node for all packages, so, when a node upgrade is required, you transfer its packages to the spare, and remove the node to be upgraded from the cluster. You are now free to upgrade the node and its software, do final testing and get project signoff without significantly impacting the highly available environment. When completed, bring the upgraded node back into the cluster, either as the new spare, or as the returned active node.

## **Documentation Package Sign Off and Testing**

As the packages and cluster are left fully configured during the upgrades, it is especially important that proper package sign off and testing takes place on the operating system and the application. This should never be carried out by the department who developed the software. Since package failure in a running cluster must be detected, testing procedures may need to be modified to test all possible failure causes and detection behaviors. This is time consuming, but if not carried out, could cause more outage than would be caused by making the modifications using a rolling upgrade.

If rolling upgrades are not performed the scenario would be to bring down the entire system, upgrade every node to the new version of software and then restart the application on all affected nodes.

To minimize planned downtime requires application designers to understand how Serviceguard works so they can design their applications to answer the questions:

- Can a new version of an application be installed without scheduled downtime? In most cases this should be possible. The most common obstacle is changing the layout of data between releases.
  - Can different revisions of an application operate within the system simultaneously?
  - If there are problems with an application, can it roll back to a previous release?

**NOTE:**

When the data layout changes between releases, a rolling upgrade may require planned downtime to convert the data to the new format. Once the upgraded software has been brought up on the converted data, make sure the package can only run on the node that has been upgraded, until the application has been upgraded on all adoptive nodes.

## 16–2. SLIDE: Rules for Rolling Upgrade

### Rules for Rolling Upgrade



- Cluster configuration files cannot be updated until all nodes are at the same version of operating system.
- All Serviceguard commands must be issued from the node with latest version of Serviceguard.
- Only two versions of Serviceguard can be implemented during a rolling upgrade.
- Only two versions of the operating system can be implemented during a rolling upgrade.
- Binary configuration files may be incompatible.
- Rolling upgrade can only be carried out on configurations that have not been modified since the last time cluster was started.
- Serviceguard cannot be removed from a node while cluster is being upgraded.
- Any new features of Serviceguard cannot be utilized until all nodes are running that version of software.
- Keep kernels consistent.
- Hardware configurations cannot be modified.

### Student Notes

The following limitations apply to rolling upgrades:

- During the rolling upgrade of your Serviceguard cluster, you should only issue Serviceguard commands (other than `cmrunnode` and `cmhaltnode`) on those nodes with the latest revision of Serviceguard. *Performing any other tasks on another node will cause inconsistent results.*
- Cluster and Package configuration cannot be modified in any way until all nodes have been upgraded.
- Hardware cannot be modified.
- None of the new features of the new version of Serviceguard can be used until all nodes have been upgraded
- Binary configuration files may be incompatible between releases of Serviceguard. Manually copy these files only where a cold install of the operating system has taken place. *Read the release notes for any scripts that must be executed with a cold install to modify package files.*

- No more than two versions of Serviceguard may be running in a cluster at any one time.
- Rolling upgrades were not designed to allow multiple versions of Serviceguard to be operating on a long term basis. It is important that all nodes in the cluster are upgraded as quickly as possible.
- Serviceguard software must not be removed (via `swremove`) from a node while the cluster is in the process of a rolling upgrade.
- Keep kernels consistent and modify any with guidance from the operating system / application provider.

**NOTE:** If you upgrade a cluster to A.11.16 or later, the `cmclnodelist` entries are automatically updated into Access Control Policies in the cluster configuration file. All non-root user-hostname pairs will be given the role of Monitor (view only).

## 16–3. SLIDE: Serviceguard Rolling Upgrades

### Serviceguard Rolling Upgrades



- Use `cmhaltnode` to halt the node you wish to upgrade.
- Turn off auto cluster configuration.
- Upgrade node.
- Apply patches.
- Turn on auto cluster configuration.
- Rejoin cluster using `cmlrunnode`.
- Repeat for all nodes in cluster.

### Student Notes

Be sure to plan sufficient system capacity / availability to allow packages to be moved between nodes without unacceptable loss of performance.

The only nodes which can issue configuration commands within a mixed cluster (eg. Serviceguard 11.15 and Serviceguard 11.17) are those with the latest version of Serviceguard.

## 16–4. SLIDE: Operating System Rolling Upgrades

### Operating System Rolling Upgrades



- Use `cmhaltnode` to halt node you wish to upgrade.
- Perform operating system upgrade.
- Patch operating system.
- Create all device files and reconfigure kernel.
- Install Serviceguard / add patches.
- Copy `/etc/cmcluster/cmclconfig` file from a node still running in cluster.
- Edit `/etc/rc.config.d/cmcluster` so `AUTOSTART_CMCLD=1`.
- Rejoin cluster with `cmrunnode`.
- Repeat for all other nodes in cluster.

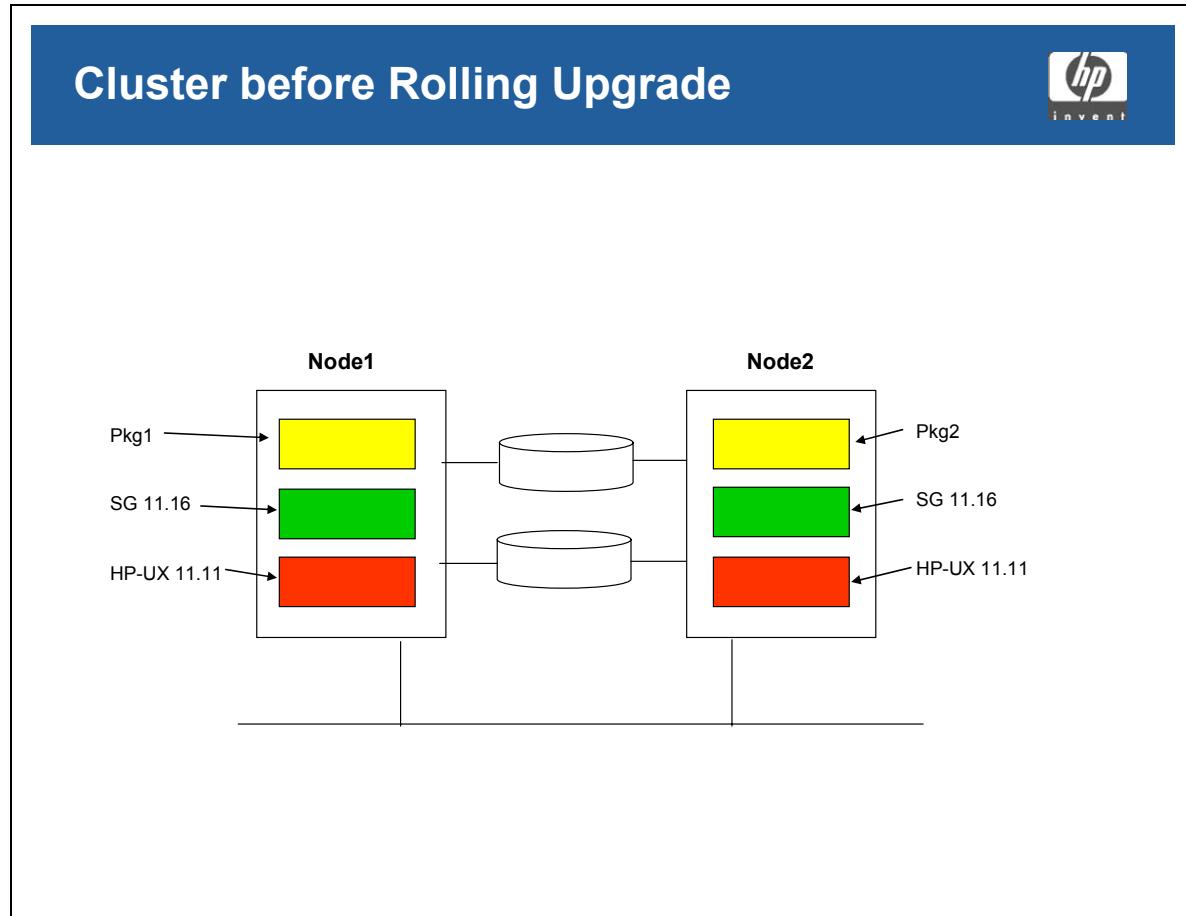
### Student Notes

Operating system upgrades are supported in a cluster (via the rolling upgrade method). However, per the Serviceguard Release Notes, a cold install of the operating system is not supported.

On a cold install (as an alternate method of updating the operating system on a node), instance numbers can (and probably will) change, which will likely cause shared disk device file consistency issues.

Note: If a cold install is necessary, be sure to print a full `ioscan` of your system prior to re-installing. Armed with this `ioscan` listing, and referring back to the High Availability Planning module (module 3), you can alter instance numbers after the re-install as necessary.

## 16–5. SLIDE: Cluster before Rolling Upgrade



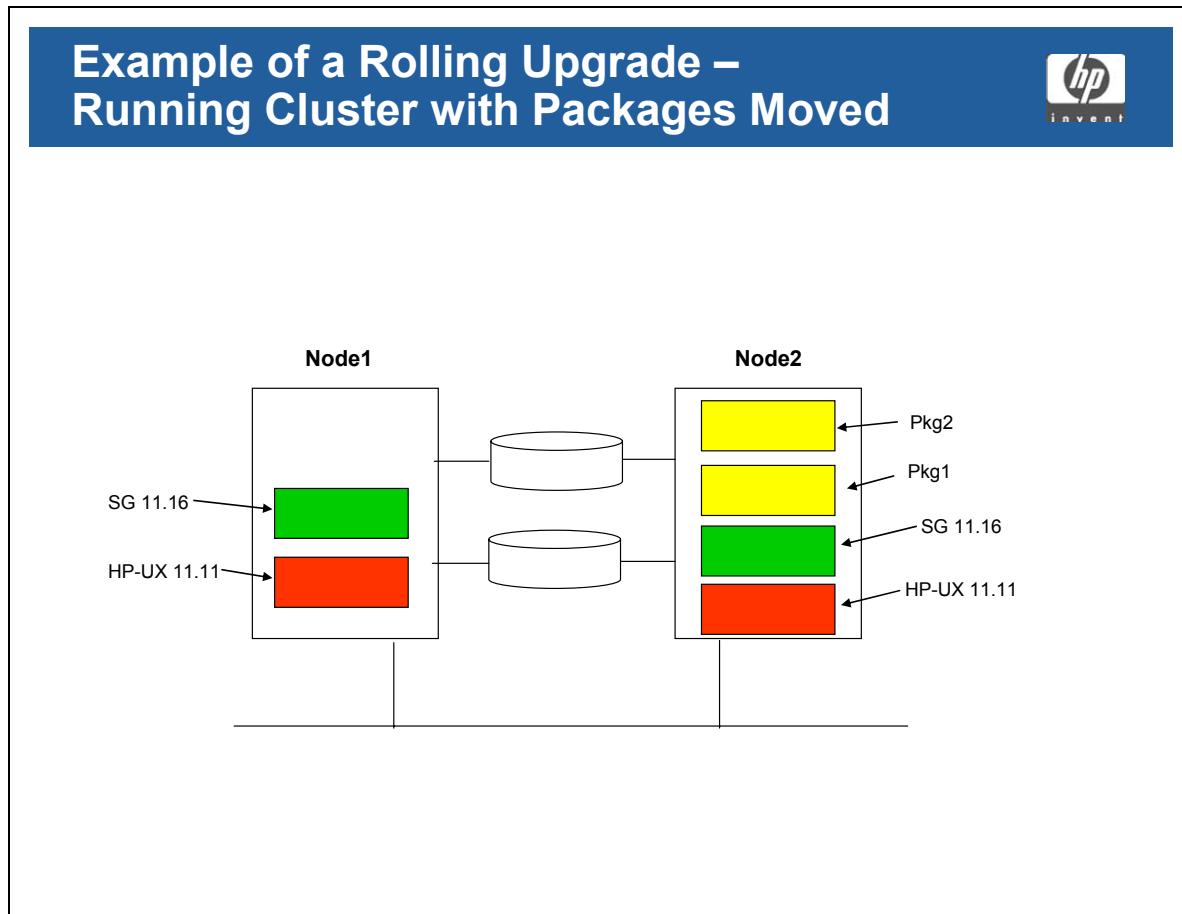
### Student Notes

- Get all kernel parameters.

```
# sysdef | lp  
# kctune | lp
```
- Get all volume groups and logical volume information.

```
# vgdisplay -v | lp  
# lvdisplay /dev/vgXX/lvolYY
```
- Halt the node.

## 16-6. SLIDE: Example of a Rolling Upgrade — Running Cluster with Packages Moved



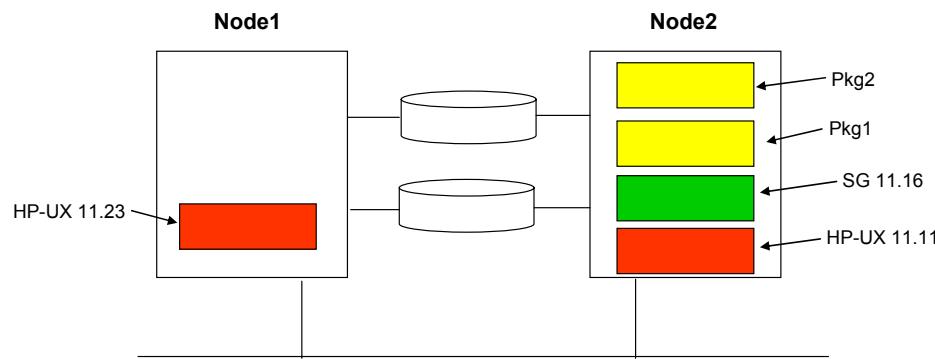
### Student Notes

All packages are up on node2.

- Backup node1.
- If this is a cold install, shutdown the machine.
- If this is an upgrade, read the release notes to identify special actions required.

## 16-7. SLIDE: Example of Rolling Upgrade — Node1 Upgraded to HP-UX 11.23

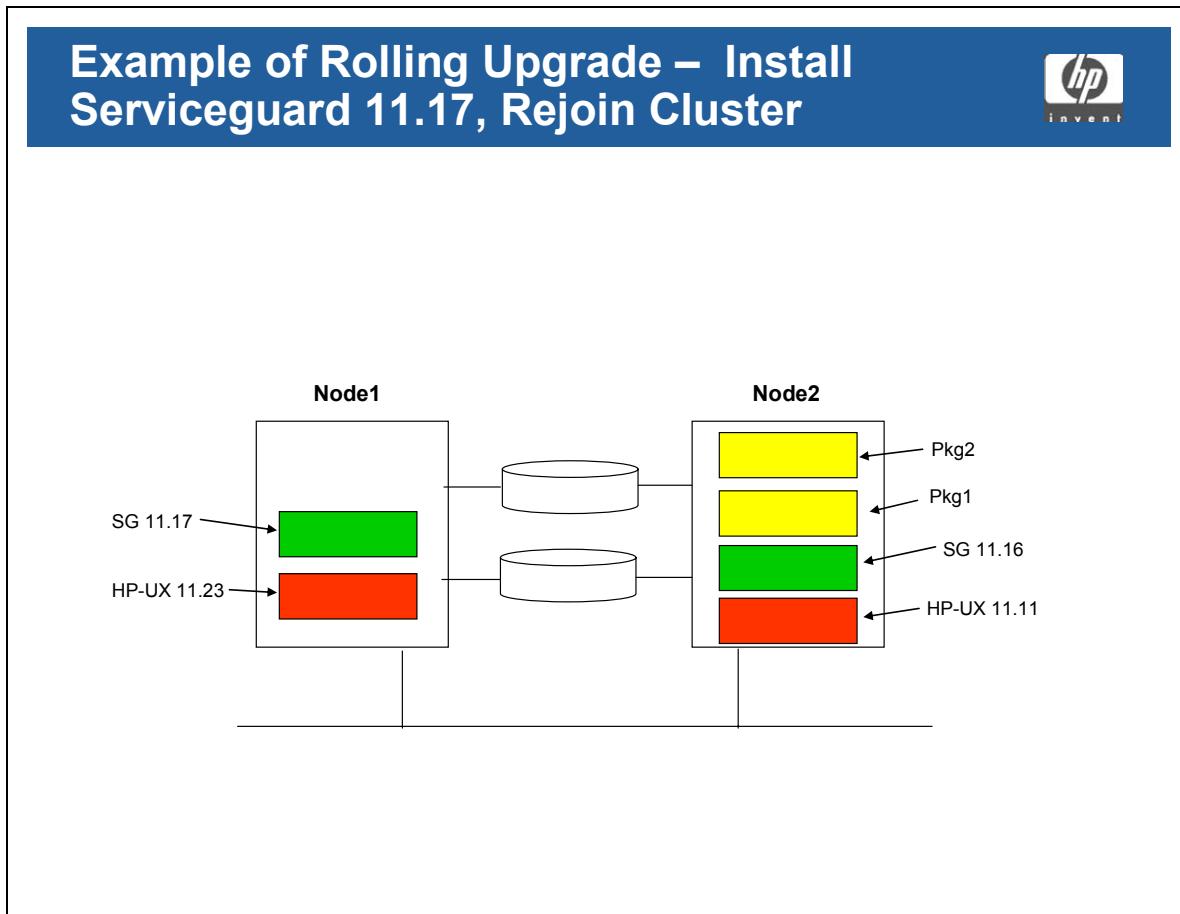
### Example of Rolling Upgrade – Node1 Upgraded to HP-UX 11.23



### Student Notes

- Update operating system.
- Patch system.
- Install correct version of Serviceguard.
- Set all kernel parameters.
- Add any drivers/subsystems to the kernel.

## 16-8. SLIDE: Example of Rolling Upgrade — Install Serviceguard 11.17, Rejoin Cluster



### Student Notes

- Upgrade any applications that require modification.

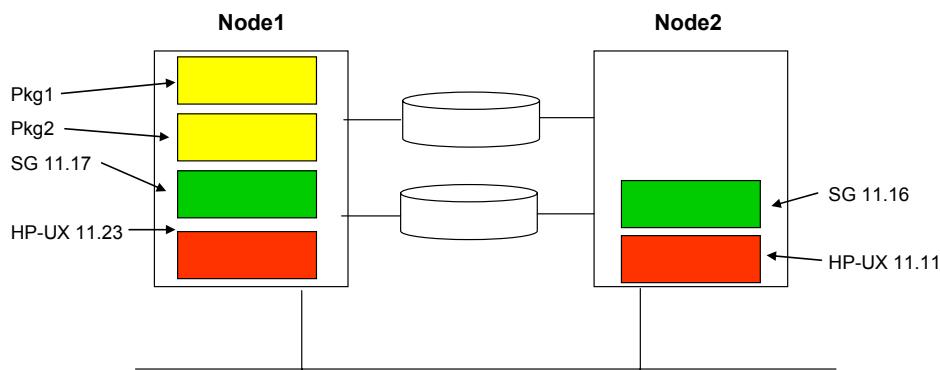
**NOTE:** All static data should be in `/opt/<app>` and all node specific data should be in `/etc/opt/<app>`.

You should be able to copy `/opt/<app>` to any node, as this should be node-independent if designed correctly.

- Use standard sign off procedures with test data to ensure the node is working correctly. You should have standard test configurations that can be applied to allow sign off of operating system and application modifications.
- Backup node.

## 16–9. SLIDE: Example of Rolling Upgrade — Run Cluster with all Packages on Node1

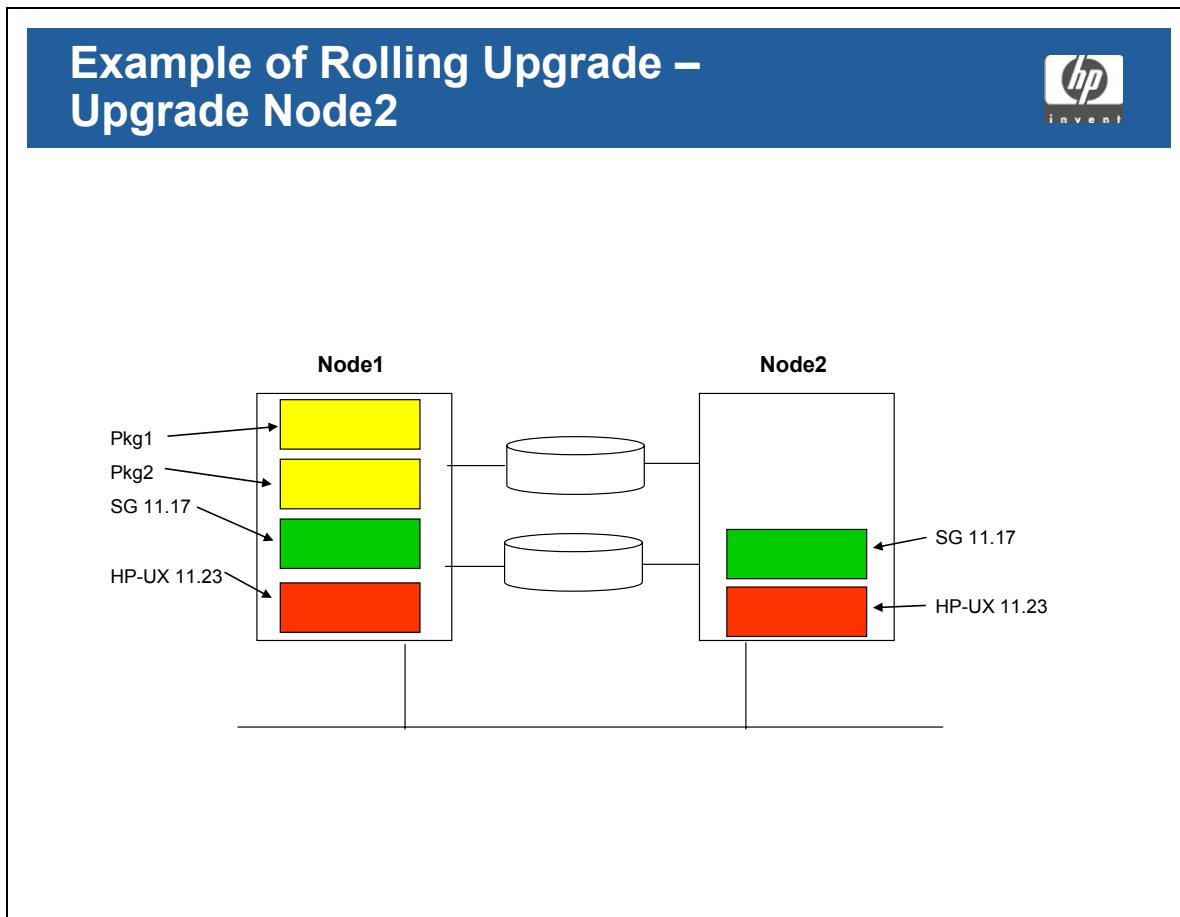
### Example of Rolling Upgrade – Run Cluster with All Packages on Node1



### Student Notes

- Add node to cluster.  
`# cmrunnode -v node1`
- Observe log files and check cluster.
- Move all packages to node1.  
`# cmhaltnode -v -n node2`

## 16-10. SLIDE: Example of Rolling Upgrade — Upgrade Node2



### Student Notes

- Upgrade operating system on node2 to HP-UX 11.23.
- Apply operating system patches as necessary.
- Install Serviceguard 11.17.
- Apply Serviceguard patches as necessary.
- Return node2 to the running cluster.

```
# cmrundnode -v <node2>
```

- Move Pkg2 back to its original node.

```
# cmhaltpkg -v Pkg2
# cmrumpkg -v -n <node2> Pkg2
```

## 16-11. SLIDE: When a Rolling Upgrade Is Not Possible

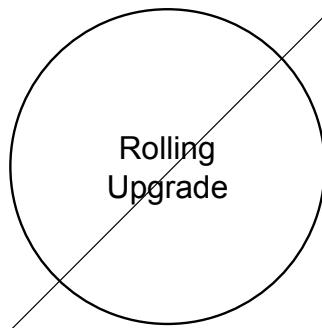
### When a Rolling Upgrade Is Not Possible



Major database conversion required

=

Rolling Upgrade



### Student Notes

If a major database conversion is required to update to the new version, a rolling upgrade is not possible because the application will need to be down for an extended period to perform the database conversion.

## **16-12. LAB: Performing a Rolling Upgrade**

### **Directions: Install the Auto-Port Aggregation Software**

#### **Special Note**

In this lab, we install the Auto-Port Aggregation software, part number J4240AA, version B.11.23.05 or later. We will not actually use this software, however. We are installing it only for the purpose of learning the "rolling upgrade" technique.

You may find that there is insufficient disk space in the /var file system when installing the Auto-Port Aggregation software. This may be the result of crash dumps created on /var/adm/crash from earlier modules. You may simply remove these crash dump files and then try installing the Auto-Port Aggregation software again.

### **Preparation for the Rolling Upgrade and Assumptions**

In this lab, we assume that the cluster is up and running, and that all packages are up and running on their respective primary nodes.

1. Write the hostname of all nodes in your cluster in the order that you wish to upgrade them.

\_\_\_\_\_  
<Node1>

\_\_\_\_\_  
<Node2>

### **Upgrade the First Node in Your Cluster**

2. Halt each package currently running on <Node1> (as noted above in step 1).
3. Next, verify that <Node1> is now up and running in the cluster, but with no packages.

Module 16  
**Rolling Upgrade Issues**

4. Now, restart each of <Node1>'s packages on its respective adoptive node.
5. Now that each package (previously halted in step 2 above) is running on its first adoptive node (step 4 above), re-enable the global package switching parameter for each of these packages. We do this so that the package can still failover if necessary.
6. Also, for each of the above packages, turn off the "local" switch for the node being upgraded (i.e., the primary node for each of the above packages). We do this to ensure that the package does not attempt to fallback to the node being upgraded while we are in the middle of the upgrade.
7. Again, verify that all packages are now running on the first adoptive nodes, AND that the node to be upgraded is running with zero packages.
8. Next, halt cluster services on the node to be upgraded (thus removing it from the cluster).
9. Again, verify that this node has been removed from the cluster.
10. Set AUTOSTART\_CMCLD to 0 in the /etc/rc.config.d/cmcluster file. This will ensure that, if the node is rebooted as part of the upgrade, the node will not rejoin the cluster after the reboot.

```
AUTOSTART_CMCLD=0
```

11. Now, do **ONE** of the following steps: either 11.a OR 11.b (but not both).
  - a. Perform the upgrade of this node by using NON-INTERACTIVE "swinstall".

```
# swinstall -s /var/spool/sw -x autoreboot=true J4240AA
```

where J4240AA is the part number for the Auto-Port Aggregation software.
  - b. OR, perform the upgrade of this node by using INTERACTIVE "swinstall".

```
# swinstall
```

    - (1) At the "Specify Source" window, accept the default hostname and default depot name (/var/spool/sw).
    - (2) Then, at the "Software Selection" window, select the "Auto-Port Aggregation J4240AA, version B.11.23.05 (or later) product.
    - (3) From this point, proceed as you would normally for any "swinstall". Note that, after the analysis phase, and as the installation has begun, you will see a message that says "The system will be rebooted as soon as installation is complete." When you see this, proceed with the installation, including the reboot.
12. After the machine has rebooted, and after logging in, check to be sure that this node has NOT rejoined the cluster.
13. Now that <Node1> has been successfully upgraded, add it back into the cluster. Verify that it is now part of the cluster.
14. Re-enable AUTOSTART\_CMCLD on this node.

AUTOSTART_CMCLD=1
-------------------

Module 16  
**Rolling Upgrade Issues**

15. Return those packages currently running on the <node2> back to this node, and reset the AUTO\_RUN and local package switching parameters (if necessary). Use `cmhalt pkg`, `cmrungpkg` and `cmmodpkg` to do this.

16. Congratulations on successfully upgrading the first node in your cluster!

17. To see if the Auto-Port Aggregation software has indeed been installed, execute the following:

```
# swlist -l bundle | grep -i auto
```

**Upgrade <Node2>**

18. Repeat steps 2 through 17 on <node2>.

19. Congratulations on successfully upgrading your entire cluster using the "rolling upgrade" technique!!

## 16-13. LAB Solution: Performing a Rolling Upgrade

### Directions: Install the Auto-Port Aggregation Software

#### Special Note

In this lab, we install the Auto-Port Aggregation software, part number J4240AA, version B.11.23.05 or later. We will not actually use this software, however. We are installing it only for the purpose of learning the "rolling upgrade" technique.

You may find that there is insufficient disk space in the /var file system when installing the Auto-Port Aggregation software. This may be the result of crash dumps created on /var/adm/crash from earlier modules. You may simply remove these crash dump files and then try installing the Auto-Port Aggregation software again.

### Preparation for the Rolling Upgrade and Assumptions

In this lab, we assume that the cluster is up and running, and that all packages are up and running on their respective primary nodes.

1. Write the hostname of all nodes in your cluster in the order that you wish to upgrade them.

\_\_\_\_\_  
<Node1>

\_\_\_\_\_  
<Node2>

### Upgrade the First Node in Your Cluster

2. Halt each package currently running on <Node1> (as noted above in step 1).

#### Answer:

```
# cmhaltpkg -v package1
# cmhaltpkg -v package2
# cmhaltpkg -v package3
```

< etc >

Module 16  
**Rolling Upgrade Issues**

3. Next, verify that <Node1> is now up and running in the cluster, but with no packages.

**Answer:**

```
# cmviewcl -v
```

4. Now, restart each of <Node1>'s packages on its respective adoptive node.

**Answer:**

```
# cmrunkpkg -v -n <first_adoptive_node> package1
# cmrunkpkg -v -n <first_adoptive_node> package2
# cmrunkpkg -v -n <first_adoptive_node> package3
```

```
< etc >
```

5. Now that each package (previously halted in step 2 above) is running on its first adoptive node (step 4 above), re-enable the global package switching parameter for each of these packages. We do this so that the package can still failover if necessary.

**Answer:**

```
# cmmmodpkg -v -e package1
# cmmmodpkg -v -e package2
# cmmmodpkg -v -e package3
```

```
< etc >
```

6. Also, for each of the above packages, turn off the "local" switch for the node being upgraded (i.e., the primary node for each of the above packages). We do this to ensure that the package does not attempt to fallback to the node being upgraded while we are in the middle of the upgrade.

**Answer:**

```
# cmmmodpkg -v -d -n < primary_node > package1
# cmmmodpkg -v -d -n < primary_node > package2
# cmmmodpkg -v -d -n < primary_node > package3
```

```
< etc >
```

7. Again, verify that all packages are now running on the first adoptive nodes, AND that the node to be upgraded is running with zero packages.

**Answer:**

```
# cmviewcl -v
```

8. Next, halt cluster services on the node to be upgraded (thus removing it from the cluster).

**Answer:**

```
# cmhaltnode -v node_to_be_upgraded
```

9. Again, verify that this node has been removed from the cluster.

**Answer:**

```
# cmviewcl -v
```

10. Set AUTOSTART\_CMCLD to 0 in the /etc/rc.config.d/cmcluster file. This will ensure that, if the node is rebooted as part of the upgrade, the node will not rejoin the cluster after the reboot.

```
AUTOSTART_CMCLD=0
```

11. Now, do ONE of the following steps: either 11.a OR 11.b (but not both).

- a. Perform the upgrade of this node by using NON-INTERACTIVE "swinstall".

```
# swinstall -s /var/spool/sw -x autoreboot=true J4240AA
```

where J4240AA is the part number for the Auto-Port Aggregation software.

- b. OR, perform the upgrade of this node by using INTERACTIVE "swinstall".

```
# swinstall
```

(4) At the "Specify Source" window, accept the default hostname and default depot name (/var/spool/sw).

(5) Then, at the "Software Selection" window, select the "Auto-Port Aggregation J4240AA, version B.11.23.05 (or later) product.

(6) From this point, proceed as you would normally for any "swinstall". Note that, after the analysis phase, and as the installation has begun, you will see a message that says "The system will be rebooted as soon as installation is complete." When you see this, proceed with the installation, including the reboot.

Module 16  
**Rolling Upgrade Issues**

12. After the machine has rebooted, and after logging in, check to be sure that this node has NOT rejoined the cluster.

**Answer:**

```
# cmviewcl -v
```

13. Now that <Node1> has been successfully upgraded, add it back into the cluster. Verify that it is now part of the cluster.

**Answer:**

```
# cmrunnode -v node_being_upgraded
```

14. Re-enable AUTOSTART\_CMCLD on this node.

```
AUTOSTART_CMCLD=1
```

15. Return those packages currently running on the <node2> back to this node, and reset the AUTO\_RUN and local package switching parameters (if necessary). Use cmhaltpkg, cmrunpkg and cmmodpkg to do this.

16. Congratulations on successfully upgrading the first node in your cluster!

17. To see if the Auto-Port Aggregation software has indeed been installed, execute the following:

**Answer:**

```
# swlist -l bundle | grep -i auto
```

## Upgrade <Node2>

18. Repeat steps 2 through 17 on <node2>.
  19. Congratulations on successfully upgrading your entire cluster using the "rolling upgrade" technique!!

Module 16  
**Rolling Upgrade Issues**

---

## **Module 17 — LVM Maintenance for Packages**

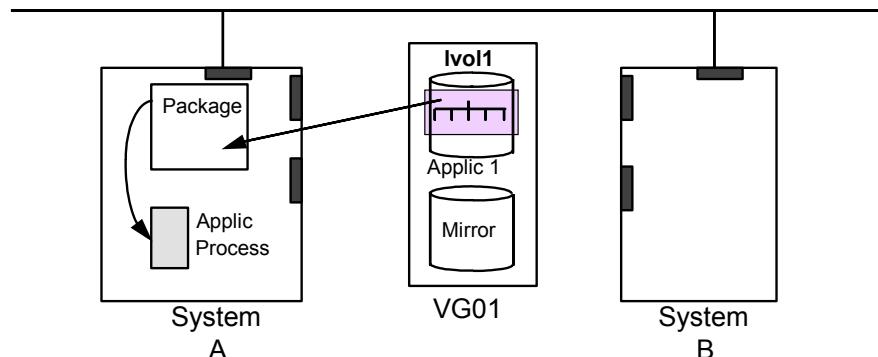
### **Objectives**

Upon completion of this module, you will be able to do the following:

- Add / remove a disk to/from a volume group owned by a package.
- Add / remove a logical volume to/from a volume group owned by a package.
- Add / remove a volume group to/from a package.
- Make a logical volume and file system (in a volume group owned by a package) larger or smaller.

## 17-1. SLIDE: Make a Logical Volume / File System Larger or Smaller

### Make a Logical Volume / File System Larger or Smaller



The /dev/vg01/lvol1 file system is full. How do we grow the logical volume so both systems recognize the new size?

### Student Notes

Using OnLine JFS, we can resize the file system using the standard tools.

If **Applic1** is running on SystemA to make the file system bigger we issue:

```
# lvextend -L newsize_in_MB /dev/vg01/sglvol1
# fsadm -F vxfs -b xxxM /sgdatal      (where xxx is in MB)
```

or to make it smaller we issue:

```
# fsadm -F vxfs -b xxxM /sgdatal      (where xxx is in MB)
# lvreduce -L newsize_in_MB /dev/vg01/sglvol1
```

**NOTE:** There is no need to issue any commands on the adoptive system.

When SystemB adopts this application the new size will be recorded in the reserved areas on the disk and will automatically be known to SystemB.

## 17-2. SLIDE: LVM Maintenance to a Package

### LVM Maintenance to a Package

**Both systems need their LVM configurations updated whenever any of the following modifications are performed:**

Add/Remove

- Disk
- Logical Volume
- Volume Group

```
graph LR; SA[SystemA] --- VG[Volume Group]; VG --- S6[Disk 6]; VG --- S5[Disk 5]; VG --- S4[Disk 4]; VG --- S3[Disk 3]; VG --- S2[Disk 2]; VG --- S1[Disk 1]; VG --- SB[SystemB];
```

### Student Notes

When multiple nodes are placed into a Serviceguard cluster, the procedure for performing a variety of administrative tasks must be modified to address the clustered environment.

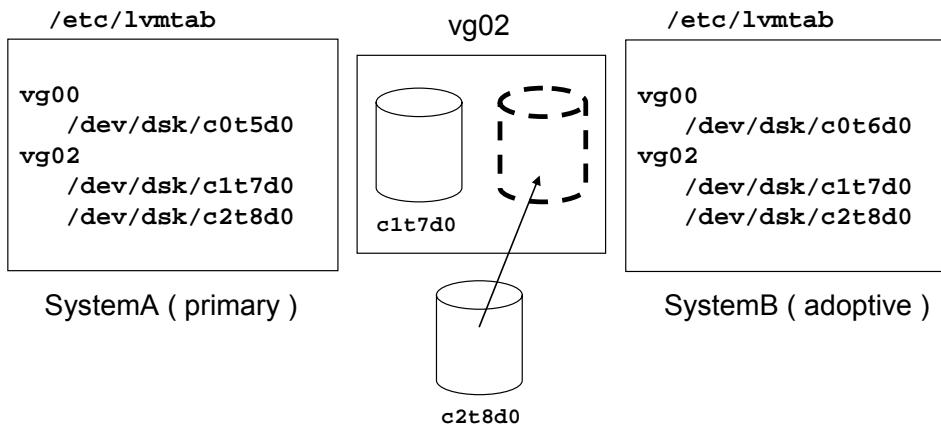
In all three cases,

- adding or removing a disk from a volume group owned by a package,
- adding or removing a logical volume from a volume group owned by a package, and
- adding or removing a volume group from a package,

changes must be made to both the primary and adoptive systems. If this is not done properly, package activation may fail on the adoptive system.

### 17-3. SLIDE: Add Disk to Volume Group Owned by a Package

#### Add Disk to Volume Group Owned by a Package



#### Student Notes

When a disk is added to a volume group with the `vgextend` command, `/etc/lvmtab` is updated with the additional disk. If we execute the `vgextend` on the primary, `/etc/lvmtab` on the primary gets updated correctly. The `/etc/lvmtab` on the adoptive system does not automatically get updated. To update `/etc/lvmtab` on the adoptive system we must re-`vgimport` this volume group after it is extended. This will require issuing a `vgexport` for it, and then recreating the volume group subdirectory in `/dev` and recreating the group file. If the map file no longer exists, we will need to recreate that also.

#### Special Note

In the following example, we are discussing volume group `vg02`. The Serviceguard administrator should bring down the package that uses `vg02`. If the administrator does not do this, and if the package should failover during the following process, then `vg02` could be left in an inconsistent state.

For example, if we are adding disk `/dev/dsk/c2t8d0` to volume group `vg02`:

- On the primary node:

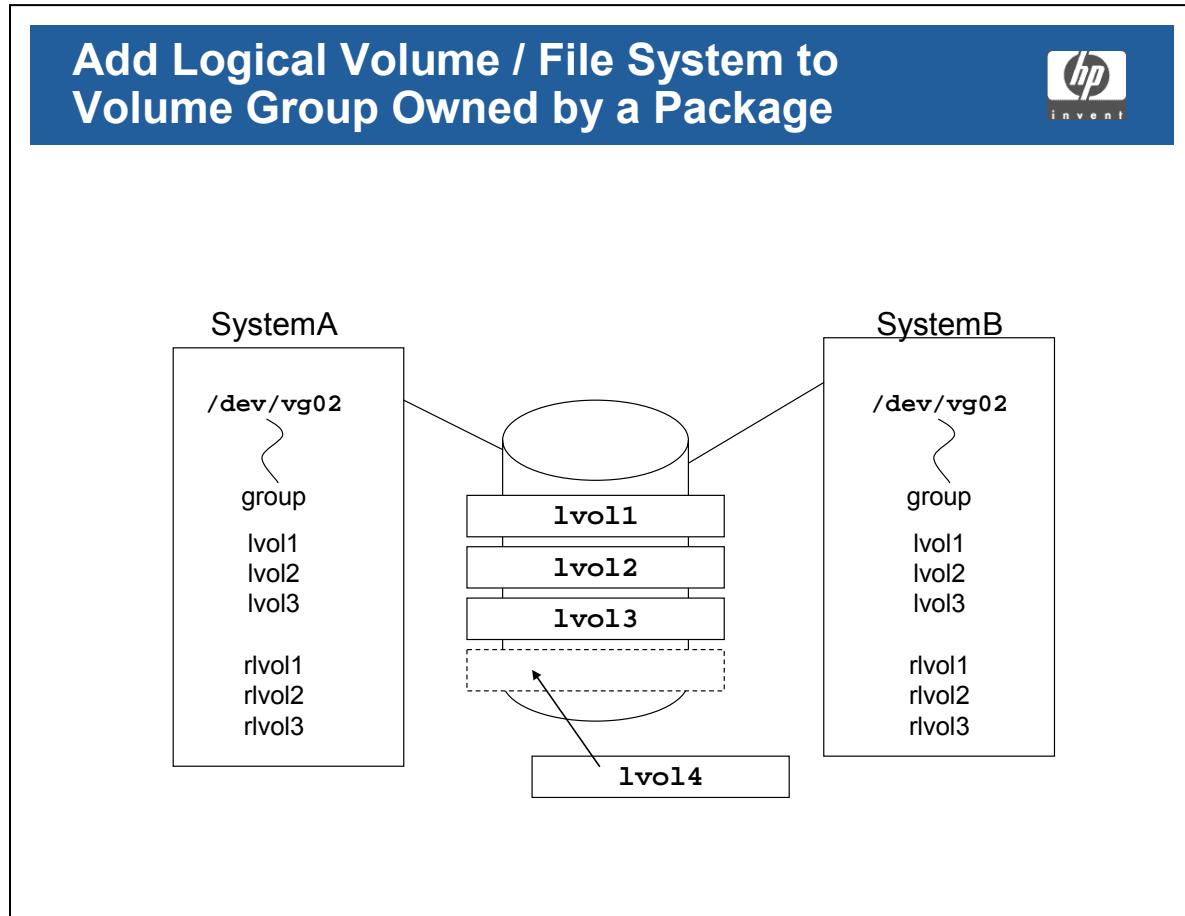
```
# cmhaltpkg -v package_name      /* The package using volume group vg02 */
# vgchange -a e vg02
# pvcreate [ -f ] /dev/rdsk/c2t8d0
# vgextend /dev/vg02 /dev/dsk/c2t8d0
# vgexport -v -p -s -m /etc/lvmconf/vg02.map vg02
# vgchange -a n vg02
```

- On the adoptive node:

```
# cd /etc/lvmconf
# rcp <primary_node>:/etc/lvmconf/vg02.map .
# vgexport -v vg02
# mkdir /dev/vg02
# mknod /dev/vg02/group c 64 0x020000
# vgimport -v -s -m /etc/lvmconf/vg02.map vg02
# vgchange -a r vg02
# vgcfgbackup vg02
# vgchange -a n vg02
```

The same issue exists when removing a disk from a volume group. The **vgreduce** command will update `/etc/lvmtab` on the primary system but not on the adoptive system. So again, we must issue a **vgimport** command for the volume group on the adoptive system after issuing a **vgreduce** command for the primary.

## 17-4. SLIDE: Add Logical Volume / File System to Volume Group Owned by a Package



### Student Notes

We can add a logical volume to a volume group with the `lvcreate` command. This command automatically creates a block and raw device file for the newly created logical volume on the primary system, but not on the adoptive system. So again, it is necessary to `re-vgimport` the volume group on the adoptive system after creating the new logical volume on the primary so that the new device files get created on the adoptive system.

Also, the package run/halt script needs to be modified if this logical volume will hold a file system, so that when the package starts this new file system is mounted and when the package halts this new file system is unmounted.

### Special Note

Once again, in the following example, we are discussing volume group `vg02`. The Serviceguard administrator should bring down the package that uses `vg02`.

If the administrator does not do this, and if the package should failover during the following process, then `vg02` could be left in an inconsistent state.

- On the primary:

```
# cmhalt pkg -v package_name
# vgchange -a e vg02
# lvcreate -L size -n lvol_name /dev/vg02
# newfs -F vxfs /dev/vg02/r1vol_name
# mkdir /new_mount_point
# vgexport -v -s -p -m /etc/lvmconf/vg02.map vg02
    (this will create a new map file that now includes new logical volume)
# vgchange -a n vg02
```

- On the adoptive:

```
# cd /etc/lvmconf
# rcp <primary_node>:/etc/lvmconf/vg02.map .
# mkdir /new_mount_point          (needed on both systems)
# vgexport -v /dev/vg02
# mkdir /dev/vg02
# mknod /dev/vg02/group c 64 0x020000
# vgimport -v -s -m /etc/lvmconf/vg02.map vg02
# vgchange -a r vg02
# vgcfgbackup vg02
# vgchange -a n vg02
```

Do not forget that we must halt the package with `cmhalt pkg`. (If the package control script needs to change it is recommended that the package be halted to avoid timing problems.)

Modify the control script on both systems.

After the scripts are changed on both systems, the package can be restarted with `cmrunk pkg`.

Re-enable package switching with `cmmodpkg -e`. (`cmhalt pkg` disables package switching)

Verify new file system has been mounted properly and is accessible on the primary.

You may want to test the adoptive system by failing the package over to the adoptive node and verifying the new file system is mounted and accessible on the adoptive node.

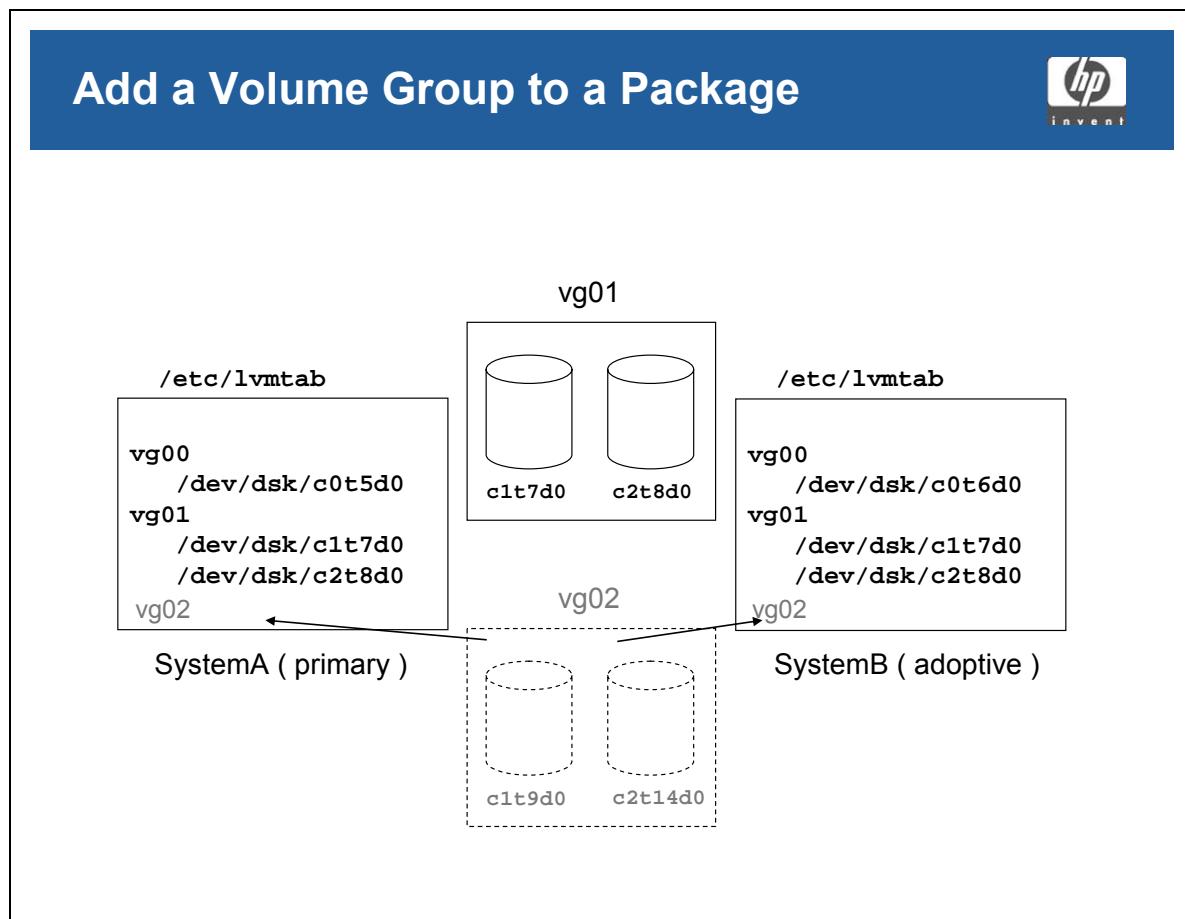
---

**NOTE:**

Removing a logical volume requires using the `lvremove` command. Again, the device files are removed on the primary but not on the adoptive node so we must re-`vgimport`. Also, the scripts must be changed so we do not try to mount / unmount the file system that has been removed. Again, the package should be halted before the control scripts are changed and then the package can be restarted after the scripts are changed on both systems.

---

## 17–5. SLIDE: Add a Volume Group to a Package



### Student Notes

When the **vgcreate** command is issued on the primary node to create a new volume group, /etc/lvmtab on the primary gets updated but /etc/lvmtab on the adoptive node does not get updated. So again it is necessary to **vgimport** on the adoptive node.

### Special Note

Once again, in the following example, we are discussing volume group vg02. The Serviceguard administrator should bring down the package that will use vg02. If the administrator does not do this, and if the package should failover during the following process, vg02 could be left in an inconsistent state.

- First, on the primary node:

```
# pvcreate [ -f ] new_disk1
# pvcreate [ -f ] new_disk2
# mkdir /dev/vg_newone
# mknod /dev/vg_newone/group c 64 0x140000 ← Note minor number
# vgcreate vg_newone /dev/dsk/new_disk1 /dev/dsk/new_disk2
```

At this point, you would create any needed logical volumes and file systems for the volume group.

```
# vgexport -v -p -s -m /etc/lvmconf/vg_newone.map vg_newone
```

- Next, on the adoptive node:

```
# cd /etc/lvmconf
# rcp <primary_node>:/etc/lvmconf/vg_newone.map .
# mkdir /dev/vg_newone
# mknod /dev/vg_newone/group c 64 0x140000
# vgimport -v -s -m /etc/lvmconf/vg_newone.map vg_newone
```

We would now create any needed mount points for file systems.

- Now, back on the primary node:

```
# vgchange -c y vg_newone
```

We then mark the new volume group as being owned by the cluster. Unmount all file systems in the new volume group, deactivate the volume group, and halt the package that will own the new volume group.

```
# vgchange -a n vg_newone
# cmhalt pkg
```

- Modify the control scripts on both systems to insure this new vg is activated when the package starts and to insure all new file systems in this new volume group are mounted.

```
# cmrunkpg -v package_name
# cmmodpkg -v -e package_name
```

The new volume group should be activated and all new file systems in this vg should be mounted and accessible where the package is running.

Again, failing over the package onto the adoptive node should verify the new vg and file systems are available on the adoptive node.

If a volume group needs to be removed from a package we can stop the package, modify the control scripts, and restart the package. This would leave the vg on the systems, but remove it from the package.

If the volume group also needs to be removed from the systems, we can issue vgexport on both systems.

## **17–6. LAB: Increasing the Size of a Logical Volume / File System**

Make the `/sgdata1` file system larger. Recall that `/sgdata1` is used by the `test` package.

## **17-7. LAB: Add and Remove an Unused Disk to Volume Group "vg01", Owned by the "test" Package**

1. Make a note of an unused disk.
  2. Make a note of the volume group owned by the **test** package.
  3. Make a note of the disks currently included in this volume group.
  4. Add the unused disk to the volume group.
  5. Create an additional mirror copy of `/dev/vg01/sg1vol1` on this newly added disk.
  6. If the package failed over right now, would the new disk and its new mirror copy be accessible on the adoptive system? Has it been added to `/etc/lvmtab` on the adoptive system? Let's find out.

Failover the package onto the adoptive system. Issue an `lvdisplay` on the logical volume with the extra mirror copy. Is the extra mirror copy current and accessible? Why not?

Module 17  
**LVM Maintenance for Packages**

7. Move package back to the primary and re-enable all package switching.
8. Re-`vgimport` the `vg01` volume group on adoptive node, recreating map file if necessary. (Remember to `vgexport` on the adoptive node first).
9. Is new disk reflected in `/etc/lvmtab` on the adoptive node?
10. Failover the package again. Issue `lvdisplay` on the new logical volume. Is the extra mirror copy current and accessible?
11. Take all steps necessary to `lvreduce` the extra mirror copy from extra disk, and remove the extra disk from the volume group on the primary.
12. Does `/etc/lvmtab` on the adoptive node still have the extra disk even though it has been removed from the volume group on the primary? What error messages would we expect to receive on the adoptive node if the package failed over? Try it. How do we eliminate these error messages?

## 17-8. LAB: Add and Remove a Logical Volume/File System to the Volume Group Owned by the "test" Package

### On the Primary Node:

1. Add a 32M logical volume named `sglvol2` to the `test` package's volume group.
  
  
  
  
  
  
2. Build a file system in this new logical volume.
  
  
  
  
  
  
3. Create a mount point, `/sgdata2`, on both systems.
  
  
  
  
  
  
4. Remember, if the package control script needs to be modified, the package should first be halted. Halt the package.
  
  
  
  
  
  
5. Modify the control script on both systems to include this new logical volume/file system.
  
  
  
  
  
  
6. Restart the package. Verify the new file system is mounted.

Module 17  
**LVM Maintenance for Packages**

7. What would happen if the package failed over right now? Does the adoptive system have device files for this new logical volume? Would the package have problems starting?

On the adoptive node:

8. Take the steps necessary on the adoptive node so that the package will failover properly. Remember, you will need a new map file that includes the newly-created logical volume.
9. Failover the package and verify the new file system is mounted and exported on the adoptive node.
10. Halt the `test` package.

On the primary node:

11. Remove the newly created logical volume on the primary and modify the control script to remove the references to this file system.
12. Restart the package on the primary and verify the package starts properly. The adoptive node now has extra device files. What do we need to do to ensure the adoptive node understands what this volume group looks like?

## 17-9. LAB: Add a New Volume Group to the "test" Package

1. On the `test` package's primary node, add a new volume group, named `vg05`, using 1 or (preferably) 2 unused disks.
  2. In your new volume group, create a logical volume named `sglvol5`, create a `vxfs` file system in the logical volume, and build a mount point at `/sgdata5`.
  3. If you used two disks in step 1 above, mirror the `sglvol5` logical volume.
  4. Halt the `test` package, modify the control script `test.cnt1` to activate this new volume group when the package starts, and mount the newly-created file system.
  5. What else needs to be done to the newly-created volume group so it can be successfully activated when the package starts up? (**Hint:** The script issues a `vgchange -a e`)  
*(If this is not done, the new volume group will not be activated)*

Module 17  
**LVM Maintenance for Packages**

6. Start the package on the primary node. Verify that the new volume group is activated and the new file system is mounted.
7. What would happen if the package attempted to start on the adoptive node at this point?
8. Take the steps necessary to define the new volume group on the adoptive node.
9. Verify that the package can failover onto the adoptive node properly.

## 17-10. LAB Solution: Increasing the Size of a Logical Volume / File System

Make the /sgdata1 file system larger. Recall that /sgdata1 is used by the test package.

### Answer:

```
# lvextend -L xxx  /dev/vg01/sglvoll      (where xxx is in MB)
# fsadm -F vxfs -b xxxM  /sgdata1        (where xxx is in MB)
```

## 17-11. LAB Solution: Add and Remove an Unused Disk to Volume Group "vg01", Owned by the "test" Package

1. Make a note of an unused disk.

**Answer:**

```
# ioscan -funC disk
# strings /etc/lvmtab
```

2. Make a note of the volume group owned by the **test** package.

**Answer:**

Review the **test.cntl** run/halt script for the volume group.

It should be **vg01**.

3. Make a note of the disks currently included in this volume group.

**Answer:**

```
# vgdisplay -v vg01
```

4. Add the unused disk to the volume group.

**Answer:**

```
# pvcreate [ -f ] /dev/rdsk/cXtYdZ
# vgextend vg01 /dev/dsk/cXtYdZ
```

5. Create an additional mirror copy of **/dev/vg01/sglvol1** on this newly added disk.

**Answer:**

```
# lvextend -m 2 /dev/vg01/sglvol1
```

6. If the package failed over right now, would the new disk and its new mirror copy be accessible on the adoptive system? Has it been added to `/etc/lvmtab` on the adoptive system? Let's find out.

Failover the package onto the adoptive system. Issue an `lvdisplay` on the logical volume with the extra mirror copy. Is the extra mirror copy current and accessible? Why not?

**Answer:**

The new disk has not been added to the `/etc/lvmtab` file on the adoptive system. Therefore, the package will not start on the adoptive node. The package will failover, but the extra mirror copy will not be accessible. The `lvdisplay` command reflects this as the extra mirror's extents should all show stale.

7. Move package back to the primary and re-enable all package switching.

**Answer:**

```
# cmhaltpkg -v test
# cmmodpkg -v -e -n <primary> test
# cmrunkg -v -n <primary> test
# cmmodpkg -v -e test
```

8. Re-vgimport the `vg01` volume group on adoptive node, recreating map file if necessary. (Remember to `vgexport` on the adoptive node first).

**Answer:**

On the primary node:

```
# vgexport -p -s -m /tmp/vg01.map vg01 ← the map file will get
                                              created even if vg01
                                              is activated.
```

On the adoptive node:

```
# vgimport vg01
# rcp <primary node>:/tmp/vg01.map /tmp/vg01.map
# mkdir /dev/vg01
# mknod /dev/vg01/group c 64 0x010000
# vgimport -v -s -m /tmp/vg01.map /dev/vg01
```

9. Is new disk reflected in `/etc/lvmtab` on the adoptive node?

**Answer:**

Yes.

Module 17  
**LVM Maintenance for Packages**

10. Failover the package again. Issue `lvdisplay` on the new logical volume. Is the extra mirror copy current and accessible?

**Answer:**

Yes.

11. Take all steps necessary to `lvreduce` the extra mirror copy from extra disk, and remove the extra disk from the volume group on the primary.

**Answer:**

On the node that the test package is currently running:

```
# lvreduce -m 1 /dev/vg01/sglvol1 /dev/dsk/cXtYdZ
# vgreduce /dev/vg01 /dev/dsk/cXtYdZ
```

12. Does `/etc/lvmtab` on the adoptive node still have the extra disk even though it has been removed from the volume group on the primary? What error messages would we expect to receive on the adoptive node if the package failed over? Try it. How do we eliminate these error messages?

**Answer:**

Does `/etc/lvmtab` on the adoptive node still have the extra disk even though it has been removed from the volume group on the primary? **Yes**  
What error messages would we expect to receive on the adoptive node if the package failed over? **Can't find disks in the VG** Try it. How do we eliminate these error messages? **You will need to clean up the `/etc/lvmtab` file on the 2<sup>nd</sup> node, as well, as follows:**

On the node on which the package is running, execute:

```
# vgexport -p -s -m /tmp/vg01.map vg01
# rcp /tmp/vg01.map <other_node>:/tmp/vg01.map
```

On the other node:

```
# vgexport vg01
# mkdir /dev/vg01
# mknod /dev/vg01/group c 64 0x010000
# vgimport -s -m /tmp/vg01.map vg01
```

## 17-12. LAB Solution: Add and Remove a Logical Volume/File System to the Volume Group Owned by the "test" Package

### On the Primary Node:

1. Add a 32M logical volume named `sglvol2` to the `test` package's volume group.

#### Answer:

```
# lvcreate -L 32 -n sglvol2 vg01
```

2. Build a file system in this new logical volume.

#### Answer:

```
# newfs -F vxfs /dev/vg01/rsglvol2
```

3. Create a mount point, `/sgdata2`, on both systems.

#### Answer:

```
# mkdir /sgdata2
```

← On both nodes

4. Remember, if the package control script needs to be modified, the package should first be halted. Halt the package.

#### Answer:

```
# cmhaltpkg -v test
```

5. Modify the control script on both systems to include this new logical volume/file system.

#### Answer:

Edit the `test.cntl` run/halt script and add:

```
LV[1]="/dev/vg01/sglvol2";FS[1]="/sgdata2"
```

6. Restart the package. Verify the new file system is mounted.

#### Answer:

```
# cmmodpkg -v -e test
# tail -f /etc/cmcluster/test/test.cntl.log
```

Module 17  
**LVM Maintenance for Packages**

7. What would happen if the package failed over right now? Does the adoptive system have device files for this new logical volume? Would the package have problems starting?

**Answer:**

The adoptive system does not have the device files `/dev/vg01/sgdata2` and `/dev/vg01/rsgdata2`. The run script will not be able to `fsck` and mount this file system. The package will have problems starting.

**On the adoptive node:**

8. Take the steps necessary on the adoptive node so that the package will failover properly. Remember, you will need a new map file that includes the newly-created logical volume.

**Answer:**

We need a new map file created **on the primary node** and copied over:

**On the primary node:**

```
# vgexport -v -p -s -m /tmp/vg01.map /dev/vg01
```

*(This will recreate the map file on the primary node.)*

**On the adoptive node:**

```
# rcp <primary_node>:/tmp/vg01.map /tmp/vg01.map
# vgexport -v vg01
# mkdir /dev/vg01
# mknod /dev/vg01/group c 64 0x010000
# vgimport -v -s -m /tmp/vg01.map vg01
# mkdir /sgdata2
```

9. Failover the package and verify the new file system is mounted and exported on the adoptive node.

**Answer:**

```
# ps -ef | grep test_program
# kill PID
```

10. Halt the `test` package.

**Answer:**

```
# cmhaltpkg test
```

**On the primary node:**

11. Remove the newly created logical volume on the primary and modify the control script to remove the references to this file system.

**Answer:**

With the package halted, the volume group should be deactivated.

```
# vgchange -a e vg01
# lvremove -f /dev/vg01/sglvol2
```

Modify run/halt script to remove references to the `sglvol2` logical volume and to the `sgdata2` file system.

12. Restart the package on the primary and verify the package starts properly. The adoptive node now has extra device files. What do we need to do to ensure the adoptive node understands what this volume group looks like?

**Answer:**

Repeat the `vgimport vg01` process on the adoptive node.

## 17-13. LAB Solution: Add a New Volume Group to the "test" Package

1. On the test package's primary node, add a new volume group, named vg05 , using 1 or (preferably) 2 unused disks.

**Answer:**

```
# mkdir /dev/vg05
# mknod /dev/vg05/group c 64 0x050000
# pvcreate [-f] /dev/rdsk/c_t_d_
# pvcreate [-f] /dev/rdsk/c_t_d_           ← If you have an extra disk
# vgcreate vg05 /dev/dsk/c_t_d_ /dev/dsk/c_t_d_
```

2. In your new volume group, create a logical volume named sglvol5 , create a vxfs file system in the logical volume, and build a mount point at /sgdata5 .

**Answer:**

```
# lvcreate -L 20 -n sglvol5 vg05
# newfs -F vxfs /dev/vg05/rsglvol5
# mkdir /sgdata5
```

3. If you used two disks in step 1 above, mirror the sglvol5 logical volume.

**Answer:**

```
# lvextend -m 1 /dev/vg05/sglvol5
```

4. Halt the test package, modify the control script test.cnt1 to activate this new volume group when the package starts, and mount the newly-created file system.

**Answer:**

```
# cmhaltpkg -v test
```

And then edit the test.cnt1 run/halt script and add:

```
VG[1]="vg05"
LV[1]="/dev/vg05/sglvol5";FS[1]="/sgdata5"
```

5. What else needs to be done to the newly-created volume group so it can be successfully activated when the package starts up? (**Hint:** The script issues a vgchange -a e) (If this is not done, the new volume group will not be activated.)

**Answer:**

```
# vgchange -c y vg05
```

6. Start the package on the primary node. Verify that the new volume group is activated and the new file system is mounted.

**Answer:**

```
# cmmodpkg -v -e test
```

7. What would happen if the package attempted to start on the adoptive node at this point?

**Answer:**

The new volume group would fail to activate and the new file system would not mount.

8. Take the steps necessary to define the new volume group on the adoptive node.

**Answer:**

Just vgimport -v vg05 on the adoptive node.

9. Verify that the package can failover onto the adoptive node properly.

**Answer:**

```
# cmviewcl -v -p test
```

Module 17  
**LVM Maintenance for Packages**

---

## **Module 18 — Serviceguard Manager**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- Discuss the components of the Serviceguard Manager.
- Navigate the Serviceguard Manager GUI.

## 18-1. SLIDE: HP Cluster Monitoring Tools

### HP Cluster Monitoring Tools



#### Tools available for monitoring / administering a Serviceguard Cluster:

- ☛ Serviceguard Manager
- ☛ EMS
- ☛ OpenView Suite
- ☛ Command line

### Student Notes

- Serviceguard Manager, an intuitive and easy to use, JAVA™-based graphical user interface.
- EMS, Event Monitoring Service provides local monitoring services for applications. We will discuss EMS in much greater detail in a later module.
- Openview is a family of network and system management products like OpenView Operations (OVO). For more information on the Openview family of products, see the "HP Openview Network Node Manager on Unix" course (HP course number B4743S). More information on this class and other Openview classes can be seen at:  
<http://www.hp.com/education/sections/network.html>.
- Command line interface. This is the traditional tool for managing a Serviceguard cluster, and is the interface we have been using in this training class all week.

## 18-2. SLIDE: Serviceguard Manager



### Student Notes

#### What Does Serviceguard Manager Provide?

Serviceguard Manager provides a visual tool to display and administer HP Serviceguard, Serviceguard SGeRAC, Metrocluster, and Continental clusters, maintaining high availability. Using Serviceguard Manager, operators see color-coded, graphically-intuitive icons to get the big-picture view of multiple clusters so that they can proactively manage the clusters, systems (nodes), and applications.

Serviceguard Manager integration with Openview (VPO 6.X, OVO 7.X and 8.0, NNM 6.X, 7.0 and 7.1) is supported on HP-UX. On XP Professional Edition, Windows 2000 and Windows 2003 Server Edition, Serviceguard Manager can be integrated with NNM 7.0, 6.31, 6.3 and 6.2.

Serviceguard Manager can be launched by HP Systems Insight Manager (versions 4.2 and 5.0) if Serviceguard Manager is installed on an HP Systems Insight Manager central management server.

Module 18  
**Serviceguard Manager**

In order to manage clusters, Serviceguard Manager must communicate with a server where Serviceguard has been installed. The server is **required** to act as a proxy to gather information from multiple clusters for Serviceguard Manager.

In order to administer clusters, a user must connect to an HP-UX server with Serviceguard version A.11.14 (or later) or a Linux server with Serviceguard version 11.14.02 (or later). If the server has Serviceguard A.11.15 (or earlier) installed, a user must connect as **root** to perform administration tasks. If the server has Serviceguard A.11.16 (or later) installed, a user can take advantage of the newly introduced role-based access capabilities, which allows non-root users to perform administration tasks. For detailed information on how to take advantage of this feature, please refer to the “**Administering Serviceguard Clusters**” section in Serviceguard Manager Online Help after Serviceguard Manager is installed.

To be discovered by Serviceguard Manager, a Serviceguard cluster needs to be on the same subnet as the server node (but not necessarily on the same subnet as the Serviceguard Manager system). Each cluster node must give access permission to the server node. Prior to Serviceguard A.11.16, such permission would be granted via the file /etc/cmcluster/cmclnodefile. With the introduction of role-based access in A.11.16, this can be achieved by configuring access control policies. For more information, see “**Before Installing Serviceguard Manager**” section in Serviceguard Manager Version A.05.00 Release Notes located at <http://www.docs.hp.com/hpux/ha>.

### **What's in Serviceguard Manager, Version A.05.00?**

- Serviceguard Manager version A.05.00 supports Serviceguard A.11.16.
- In Serviceguard Manager A.04.00, you can create or modify configuration of clusters and packages on nodes with Serviceguard version A.11.16 and newer.

Many of your configuration options are discovered by Serviceguard and displayed in lists for you to select. The configuration file and control script can be automatically created, then distributed to the nodes by clicking a button.

- Serviceguard Manager uses Access Control Policies to allow non-root access to administer clusters and packages.
- The Properties have several new tabs, including the ability to display information about physical volumes, volume groups, EMS resources, and access control policies (roles).
- The new Operations Log window shows messages for administration and configuration actions. Messages from all operations are integrated into the one screen.
- If you connect to a session server with Serviceguard A.11.16 or later, it can discover these Serviceguard Extension for RAC clusters:
  - SGeRAC 11.14 clusters with Oracle 8i or Oracle 9i
  - SGeRAC 11.15 clusters with Oracle 8i, Oracle 9i, or Oracle 10g
  - SGeRAC 11.16 with Oracle 9i or Oracle 10g

## **Supported Environments**

Nodes with the following Serviceguard versions can be servers and do the discovering:

- Serviceguard Version A.11.12 or later
- Serviceguard OPS Edition, Version A.11.12 or later
- Serviceguard SGeRAC, any version.

## **Managed Nodes**

Cluster objects with the following Serviceguard versions can be discovered.

- Serviceguard version A.10.10 or later
- Serviceguard OPS Edition A.11.01 or later
- Serviceguard SGeRAC ( all versions )
- Metrocluster ( all versions )
- Continentalclusters ( all versions )
- At least one SG Cluster Object Manager ( installed with Serviceguard version A.11.12, Serviceguard OPS version A.11.12, or Serviceguard SGeRAC )

## **Serviceguard Management Consoles**

You can install Serviceguard Manager on either a Windows PC or an HP-UX system.

The sections below outline the recommended hardware and software for running Serviceguard Manager on a Windows PC or an HP-UX system. These recommendations should be treated as **minimum** requirements for reliable operations at acceptable performance levels. It is possible to run with fewer resources at the risk of reduced performance and possibly incorrect operations.

### **HP-UX management station requirements:**

- HP-UX 11.x operating system
- 256 MB of available memory
- 190 MB of available hard disk space under /opt
- 1 MB of available hard disk space under /usr
- 1 MB of available hard disk space under /etc/opt/ov if Openview has been installed
- 3-15 MB of available hard disk space under /var for log files

**PC management station requirements:**

- Windows XP Professional or Windows 2000 Professional (with SP1 or later) or Windows 2003 Server Edition
  - Pentium II 200 MHz or higher
  - 256 MB of memory
  - 65 MB of available hard disk space
  - 1 MB of additional hard disk space should be available for log files)
  - SVGA or higher-resolution video adapter
- Installing Serviceguard Manager

If a previous version of Serviceguard Manager is already installed, you must uninstall it as directed in the next section before installing the newer version.

**Installation Instructions**

If a previous version of Serviceguard Manager is already installed, you must uninstall it before installing the new version. You can install Serviceguard Manager on a Windows, Linux, or HP-UX system. For additional information regarding Serviceguard Manager release A.05.00, go to <http://www.docs.hp.com/hpux/ha> and look for Serviceguard Manager Version A.05.00 Release Notes.

**Installing Serviceguard Manager on a Windows PC**

On a Windows PC, you can install from the download file as outlined below:

1. If a version of Serviceguard Manager earlier than A.03.00.01 has been installed on your system, you must uninstall it prior to installing this version (A.05.00). You can do this through the Windows Start menu by selecting the Start > Programs > Serviceguard Manager > Uninstall option. After you have uninstalled Serviceguard Manager, you may continue with the next step to install the latest version.
2. Click on the link under download software (sgmanager.exe) to access the self-extracting zip file used to install Serviceguard Manager.
3. You may choose to store this file on your system or run it directly from the website. If you download to your system, this file can be removed after installation.
4. Run sgmanager.exe.
5. The installation process will ask you to select a language for installation.
6. The installation process will ask for an installation directory. If you accept the default, Serviceguard Manager will be installed in:

C:\Program Files\Hewlett-Packard\Serviceguard Manager

You are now ready to run Serviceguard Manager.

## **Installing Serviceguard Manager on a HP-UX System**

On a HP-UX system, you can install from the download file as outlined below:

1. Select your operating system from the “Software Specification” pull-down list. (To see the version of an HP-UX node, use the uname command from the node.)
2. Download and store this file on your system in a directory of your choice (for example, /tmp). Do not attempt to run this file from the website. This file can be removed after installation is completed.
3. If a version of Serviceguard Manager earlier than A.03.00.01 has been installed, run the command `swremove B8325BA` to remove it. Otherwise, go to the next step.
4. Run the command `swinstall -s /tmp/B8325BA_11.X.dep` to start the installation process.
5. Follow the directions presented by `swinstall` to mark, analyze, and install Serviceguard Manager. Serviceguard Manager will be installed under the `/opt/sgmgr` directory. Also, the `/var/opt/sgmgr` directory will be created for the storage of log files.

When Serviceguard Manager is installed on a HP-UX system, the installation process examines the system to see if Openview is present. If so, the installation process adds the HP Serviceguard Manager Launcher command to the Openview menu.

Serviceguard Manager installation process then checks to see if the OVO toolkit and certain tools are installed. If these tools are not present, the installation process automatically installs them. The Serviceguard Manager installation process will not replace any tools that have already been installed.

You are now ready to run Serviceguard Manager.

### **On the managed HP-UX Serviceguard nodes:**

You should install SNMP master agent patch PHSS\_27858 for HP-UX 11.x to receive SNMP traps properly.

For Serviceguard 11.16 nodes that will be used as proxy servers for Serviceguard Manager, install PHSS\_31077 for HP-UX 11.11 or PHSS\_31078 for HP-UX 11.23.

## **Complete Installation of Serviceguard Manager on an HP-UX System**

- 1) Select the product or products to install from the bundle. This graphical user interface product is:

B8325BA              Serviceguard Manager

You may choose to also install:

B3935BA              Serviceguard  
B8324BA              HP Cluster Object Manager

Follow the directions presented by the `swinstall` GUI to mark, analyze, and install Serviceguard Manager.

When Serviceguard Manager is installed on an HP-UX system, the installation process examines the system to see if Openview is present. If so, the installation process adds the HP Serviceguard Manager Launcher command to the Network Node Manager (NNM) and/or the

Module 18  
**Serviceguard Manager**

Information Technology Operations (ITO) menu.

The Serviceguard Manager installation process then checks to see if the ITO toolkit is installed, and if certain tools are installed. If these tools are not present, the installation process automatically installs them. The Serviceguard Manager installation process will not replace any tools that have already been installed.

More information can be found at: /opt/sgmgr/OV/Readme.txt

## **Uninstalling Serviceguard Manager**

### **Uninstalling Serviceguard Manager on a Windows PC**

On a Windows system, there are two ways to remove Serviceguard Manager from your computer:

- From the Start menu, choose:  
Programs -> Serviceguard Manager -> Uninstall
- From the Start menu, choose:  
Settings -> Control Panel -> Add/Remove Programs
- Select Serviceguard Manager when the dialog box opens.

### **Uninstalling Serviceguard Manager on a HP-UX System**

On an HP-UX system, run:

```
swremove B8325BA
```

## **Running Serviceguard Manager**

### **Serviceguard Manager in Operation**

When Serviceguard Manager is run to monitor the operation of clusters, nodes and packages in real-time, Serviceguard Manager will open a live connection to a Serviceguard Object Manager server. It is the Object Manager program, running on a Serviceguard node, which in turn creates connections to the other Serviceguard nodes to gather the data presented by Serviceguard Manager.

The ability of Serviceguard Manager to monitor your systems depends on the ability of the Object Manager to connect to your systems and on the user's ability to log into the Object Manager.

## **Security Prerequisites**

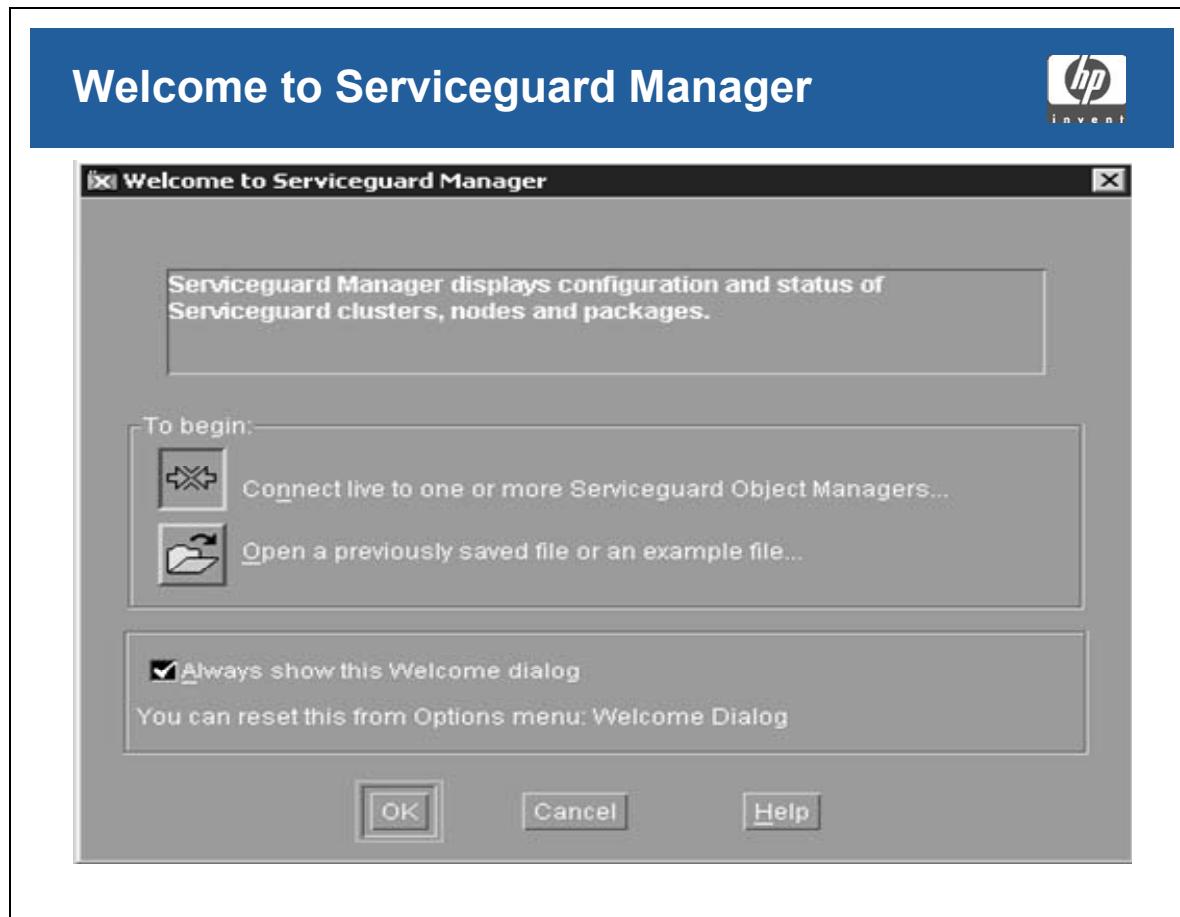
Before Serviceguard Manager is run, confirm that the security configuration of your nodes and clusters is correct. If you do not do this, Serviceguard Manager will run, but any incorrectly configured nodes and clusters will not be discovered.

The nodes in the clusters you wish to monitor must authorize the Object Manager host before it can access them for cluster information. This must be done on every node. Please refer to the *Managing Serviceguard* manual for more information. This manual can be found on the Web at:

<http://www.docs.hp.com/hpux/ha>

For more information, please see the `sgmgr(1M)` man page on any HP-UX system on which Serviceguard version A.11.12 or later has been installed.

### 18–3. SLIDE: Welcome to Serviceguard Manager



### Student Notes

#### Serviceguard Manager Startup (Platform Independent)

After Serviceguard Manager is launched, a splash screen is displayed that presents the progress of Serviceguard Manager during its initialization.

After initialization, you will see the Welcome dialog box. You can connect to a server to see a snapshot of your clusters and their status. You can update this snapshot at any time by clicking Refresh on the toolbar. You can also look at a saved object (.sgm) file.

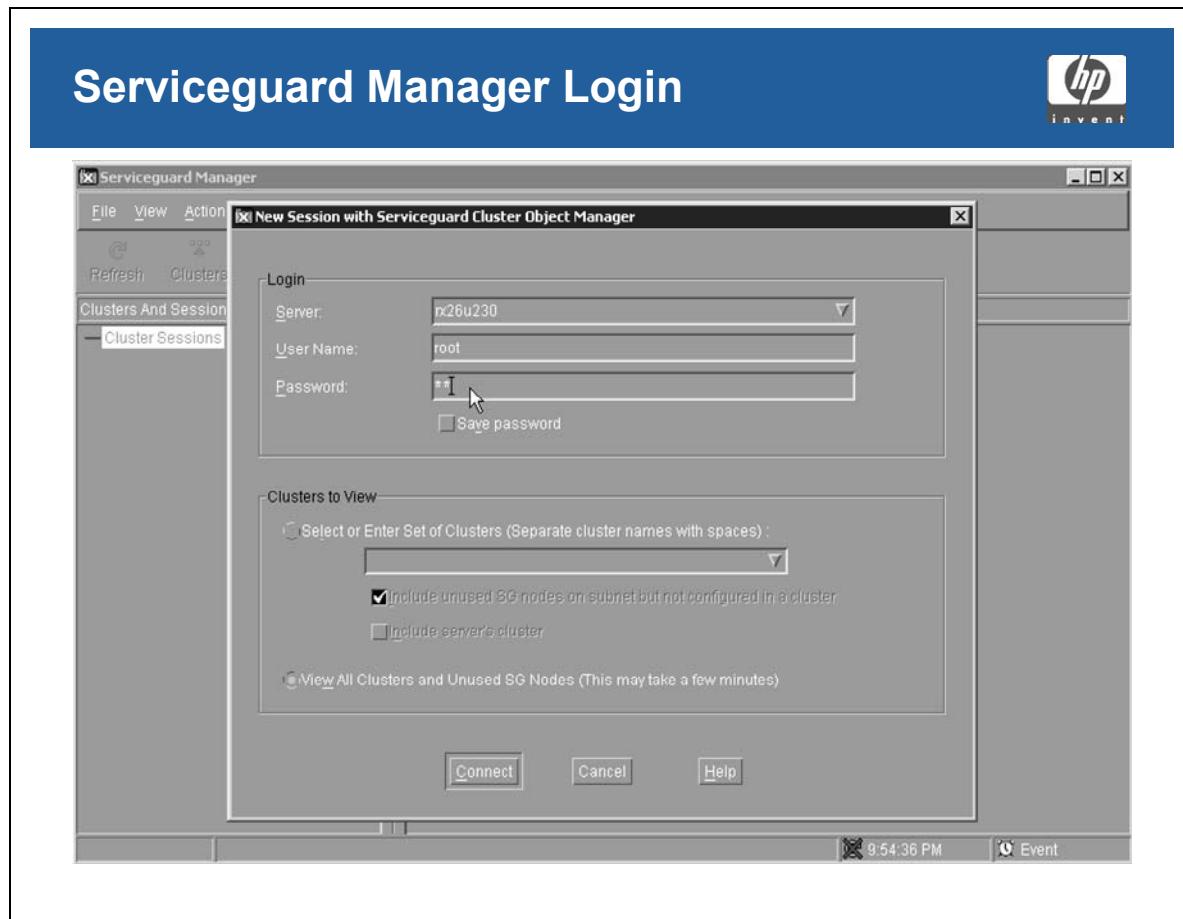
To connect to a server and display status, Serviceguard Manager must complete a discovery process that may take a significant amount of time. Once this one-time delay is incurred at startup, subsequent refresh operations are much faster.

- For a connection, choose the connect option and fill in the dialog box, specifying the host, user name and password. The server must be a node with Serviceguard version A.11.12 or later. It will discover the clusters on its subnet, and provide information to your management station. To access the connection feature later, select New Session from the File menu.

- To see a saved object (.sgm) file, choose “Open a previously saved file...” and select from the list of files presented in the dialog box. You can save maps as .sgm files from the Serviceguard Manager File menu.

To open a saved object file later, select **File -> Open**.

## 18-4. SLIDE: Serviceguard Manager Login



### Student Notes

#### Running Serviceguard Manager on a Windows PC

On a Windows system, there are four ways to launch Serviceguard Manager:

- Double-click the **SGMgr** icon on your desktop.
- From the PC Start menu, choose:

Programs -> Serviceguard Manager -> SGMgr

- From the My Computer icon or from Windows Explorer, go to:

C:\Program Files\Hewlett-Packard\Serviceguard Manager\bin

Double-click on **SGMgr** (or **SGMgr.exe**, if common extensions are not hidden).

- From a DOS window, use the **SGMgrDos** command in the directory shown above. Optional syntax at the command line is shown below.

Running from a DOS window allows the command line options to be easily entered at run-time. The command line options may also be specified from the `SGMgr` icon by changing the properties of the shortcut used to launch Serviceguard Manager.

Right-click on the icon and select `Properties`. Under the `Shortcut` tab, there is a parameter `Target`. The initial value of this parameter is probably:

```
C:\Program Files\Hewlett-Packard\SGManager\bin\SGMgr.exe
```

Note that the path name of `SGMgr.exe` is enclosed in quotes. This is good practice on a Windows PC, where the path name may include spaces, and in this case is required because of the embedded blank in the `Program Files` folder name. The quotes prevent the target of the shortcut from being interpreted incorrectly as a target program with one parameter. You may add any desired run-string parameters to the right of the closing double quote.

For example:

```
"C:\Program Files\Hewlett-Packard\SGManager\bin\SGMgr.exe" -c clus1 -c clus2
```

will limit the scope of a live connection from Serviceguard Manager (when it is launched by double-clicking on the icon) to clusters `clus1` and `clus2`.

## **Running Serviceguard Manager on a HP-UX System**

On a HP-UX system, you may launch Serviceguard Manager by using the following sequence of commands:

```
export DISPLAY=<hostname>:0.0      (if needed)
/opt/sgmgr/bin/sgmgr <options>
```

where `<hostname>` is the domain name of the system where the graphical user interface is to be displayed.

See section Command Line Syntax for an explanation of the command line options.

## **Command Line Syntax**

The Serviceguard Manager program may be executed from the command line in several ways:

- `sgmgr` — The HP-UX executable.
- `SGMgr` ( or `SGMgr.exe` ) — from the PC Desktop Icon
- `SGMgrDos` ( or `SGMgrDos.exe` ) — from a DOS window on the PC

For purposes of the following discussion, `sgmgr` will be used. There are no platform specific limitations or extensions of the command line parameters.

Serviceguard Manager provides the ability to examine live cluster data retrieved from a Serviceguard Object Manager. Command line options may be used to identify the Object Manager system, user, and password as shown below:

```
# sgmgr -s server -l user -p password
```

Module 18  
**Serviceguard Manager**

If these parameters are not supplied in the command line, the user will be asked to supply them in a connection dialog box.

By default, the live connection will display all clusters visible to the Object Manager. If a cluster does not appear, you will need to configure the security files on cluster nodes.

To increase performance and simplify activities, you can limit the scope of this discovery to a few chosen clusters, as shown below:

```
# sgmgr -c cluster1 -c cluster2 ...
```

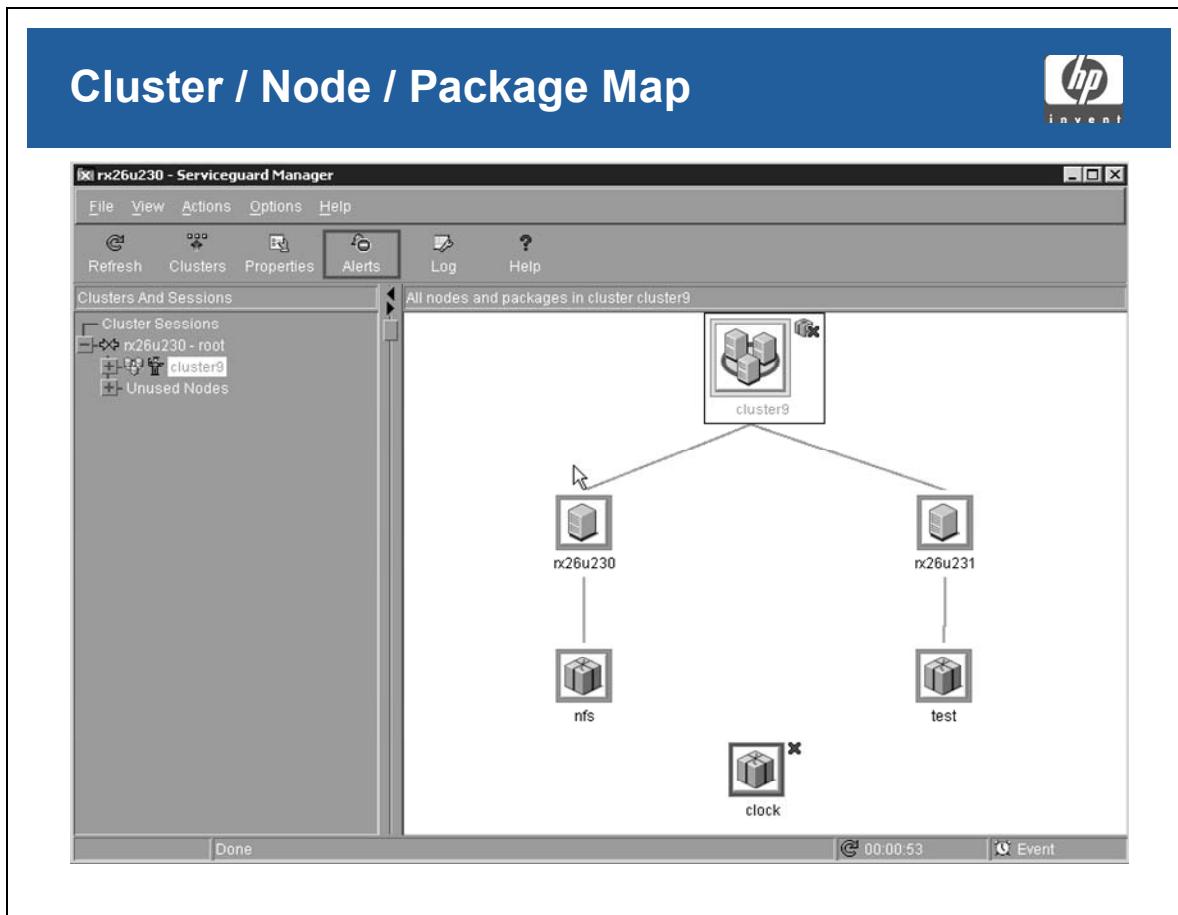
Since these parameters to limit the scope make sense only during a live connection, they can be used on the command line after the parameters for establishing the connection to the Object Manager, as shown below.

```
# sgmgr -s server -l user -p password -c cluster1 -c cluster2 ...
```

Serviceguard Manager can also view cluster data that has been saved to a file. The name of this file can be specified in the command line as shown below:

```
# sgmgr -f filename
```

## 18-5. SLIDE: Cluster / Node / Package Map

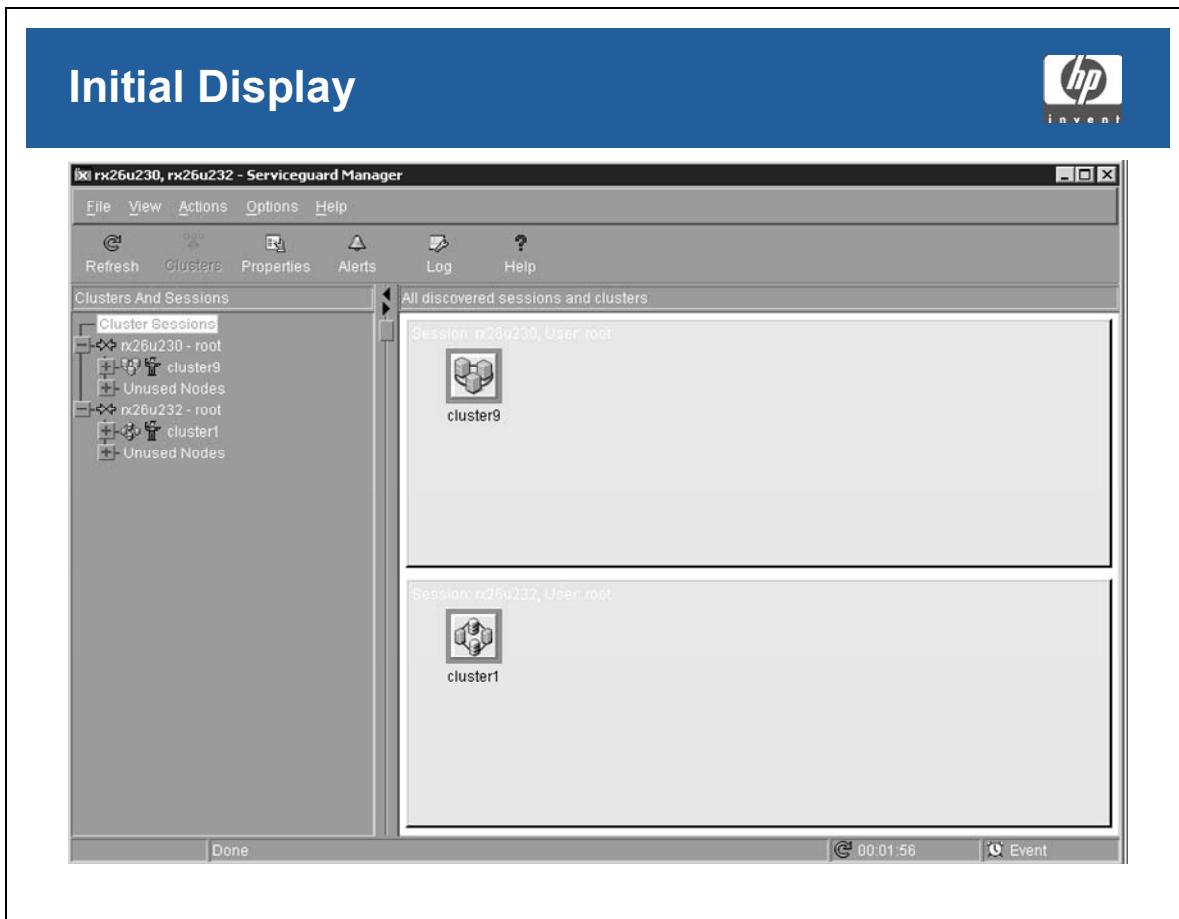


### Student Notes

The clusters are shown in a two-paned window. The left pane shows a hierarchical tree view of the clusters and their components. The right pane shows them as a map. You can change the focus of the map by:

- Selecting a different element from the tree, using the mouse (single-clicking on the left button), or by using the arrow keys.
- Double-clicking a symbol on the map.

## 18–6. SLIDE: Initial Display

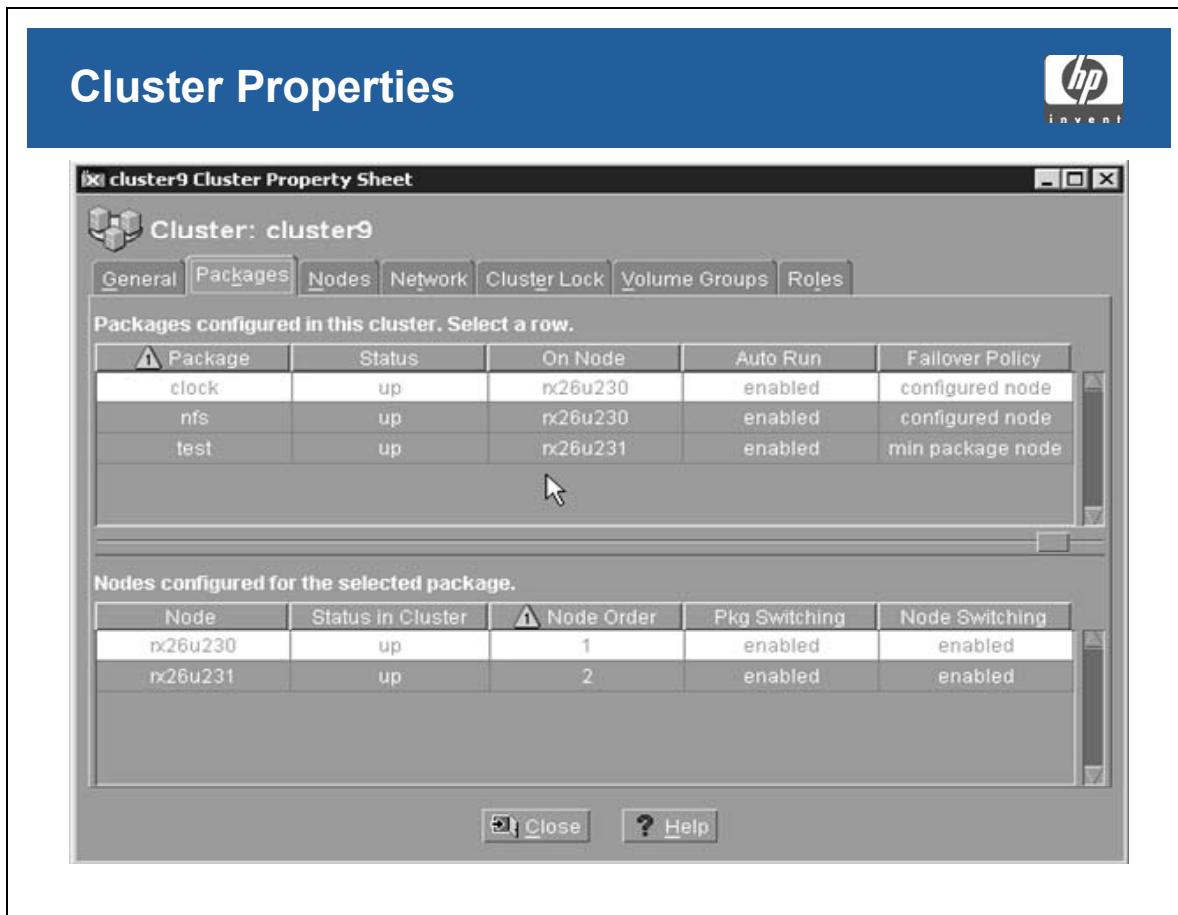


### Student Notes

After starting Serviceguard Manager, and upon login and cluster specification, the initial display shows the cluster(s) icon(s) in the right side of the display.

More cluster-specific information can be determined by either left single-clicking a cluster icon and selecting properties from the toolbar –or- by right-clicking the cluster icon and selecting properties in the right-click menu. Note that it is also possible to halt the cluster and/or make cluster modifications via the right-click menu.

## 18-7. SLIDE: Cluster Properties



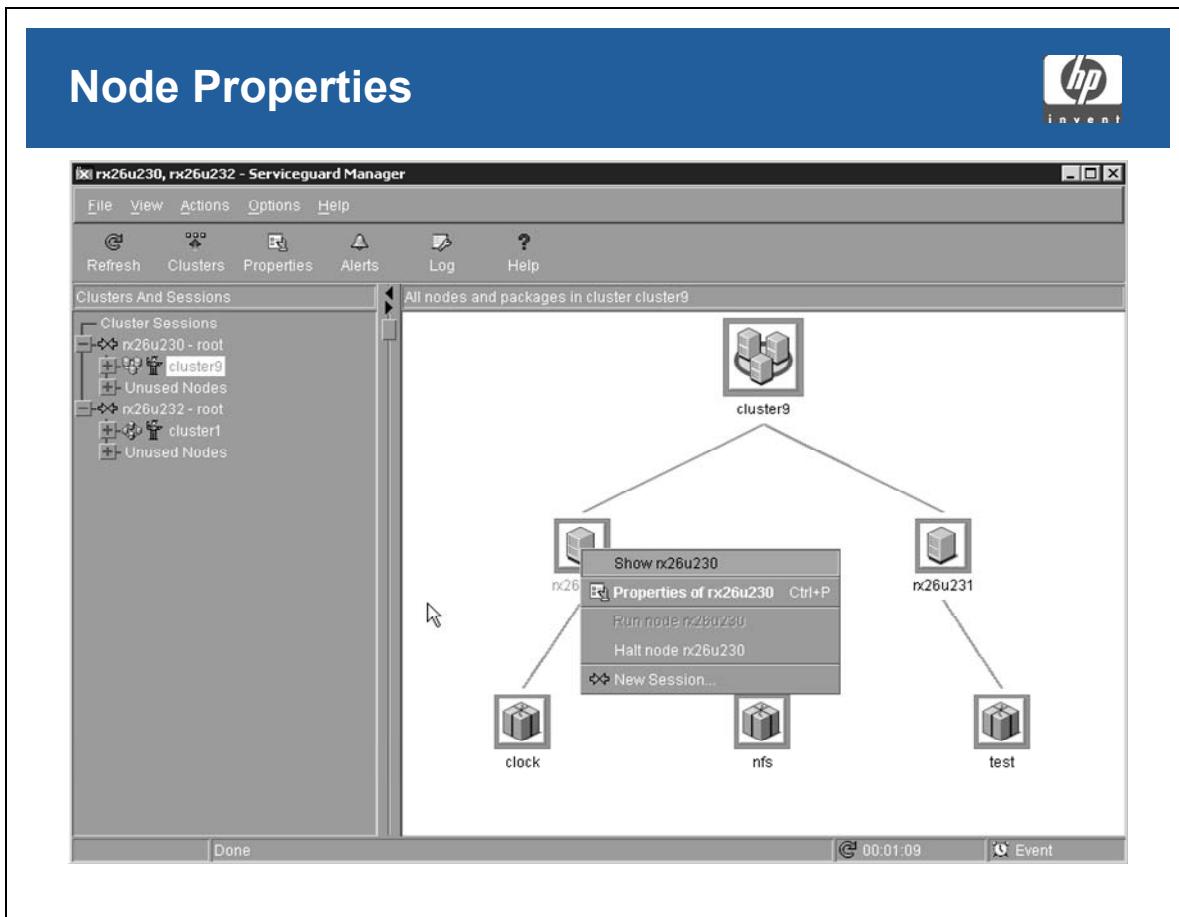
### Student Notes

Right click on the cluster icon and select Properties to view the cluster Property Sheet.

The Cluster Property Sheet has seven tabs:

- General
- Packages
- Nodes
- Network
- Cluster Lock
- Volume Groups
- Roles

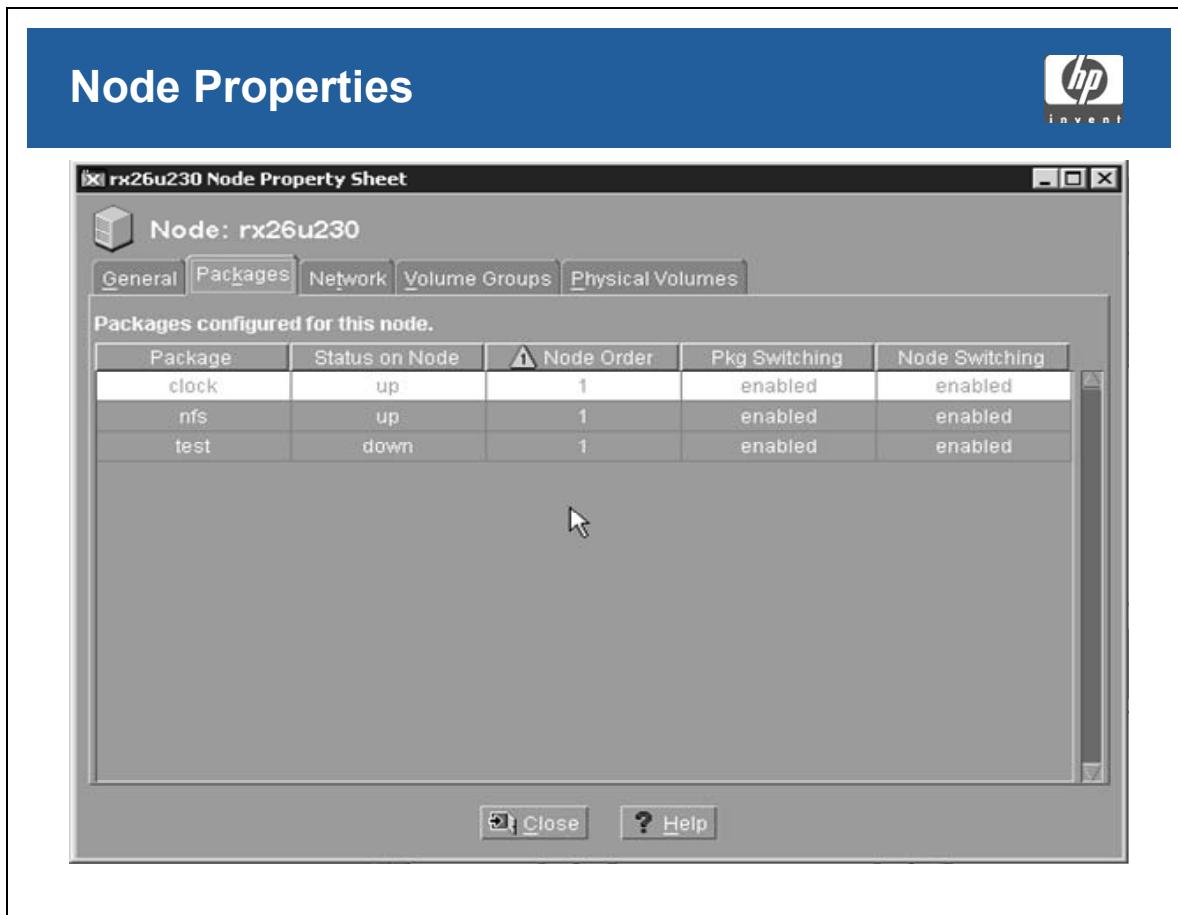
## 18-8. SLIDE: Node Properties



### Student Notes

Double-clicking the cluster icon will expand the view of the cluster. Right-clicking on a node will allow the operator to either view the node properties or halt the node.

## 18-9. SLIDE: Node Properties



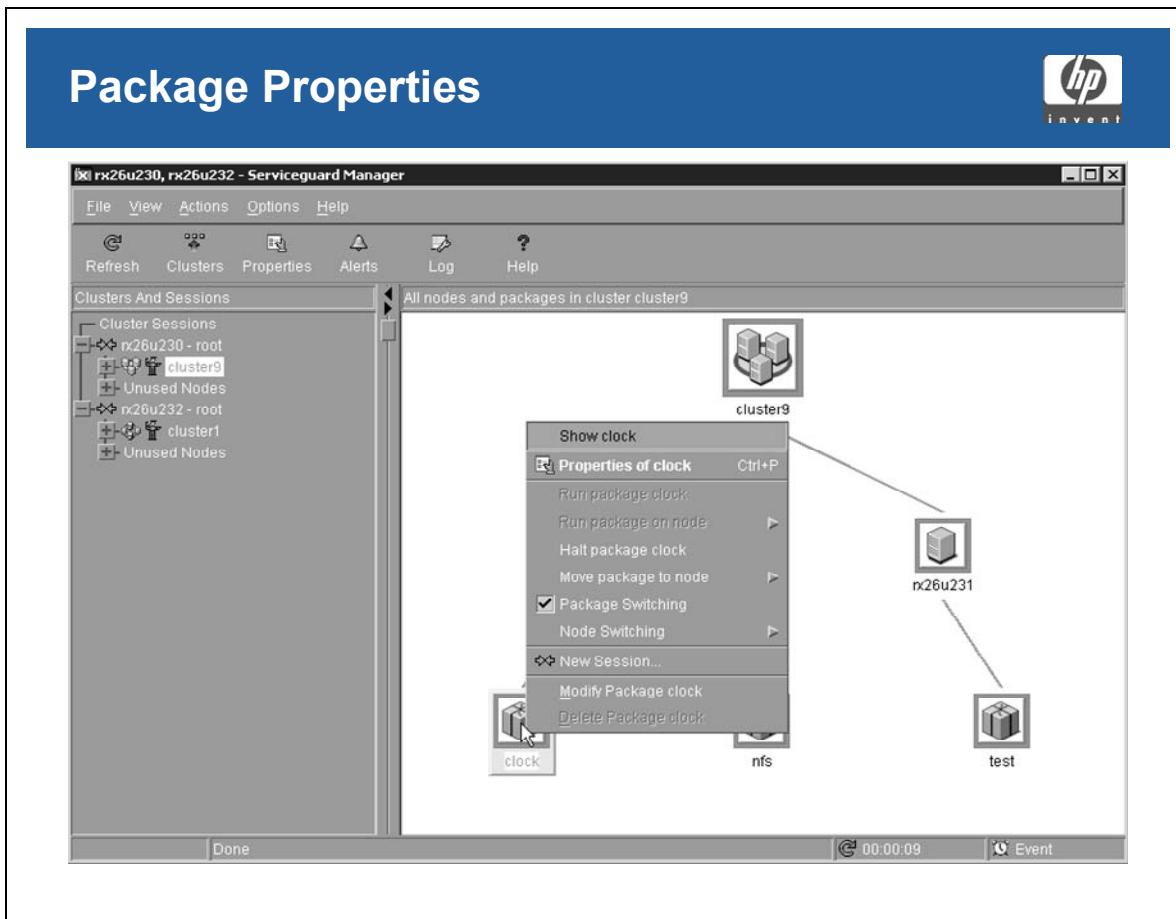
### Student Notes

After expanding the cluster (by double-clicking the cluster icon), more information is available. For example, right-clicking on a node allows a view of the node properties.

The Node Property Sheet has six tabs:

- General
- Packages
- Network
- Volume Groups
- Physical Volumes

## 18-10. SLIDE: Package Properties



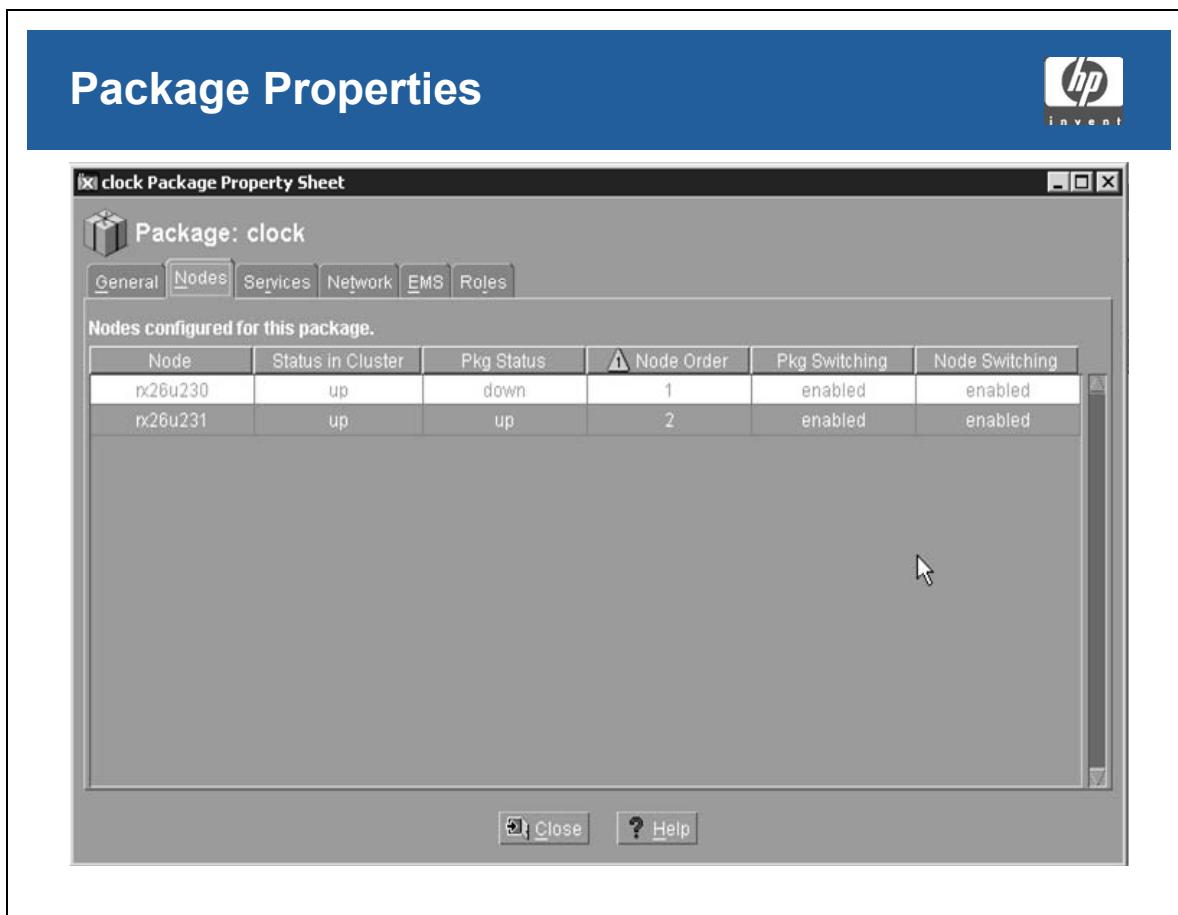
### Student Notes

Right click on a package icon and select **Properties** to view the Package Property Sheet.

The Package Property Sheet has six tabs:

- General
- Nodes
- Services
- Network
- EMS
- Roles

## 18-11. SLIDE: Package Properties



### Student Notes

Right-clicking on a package icon allows a view of the package properties and allows several actions. From the right-click menu, it is also possible to do the following:

- Halt the package
- Move the package (which halts the package on the current node and starts it up again on an alternate node)
- Enable/disable package switching
- Enable/disable node switching

NOTE: It is also possible to click and drag a package from one node to another.

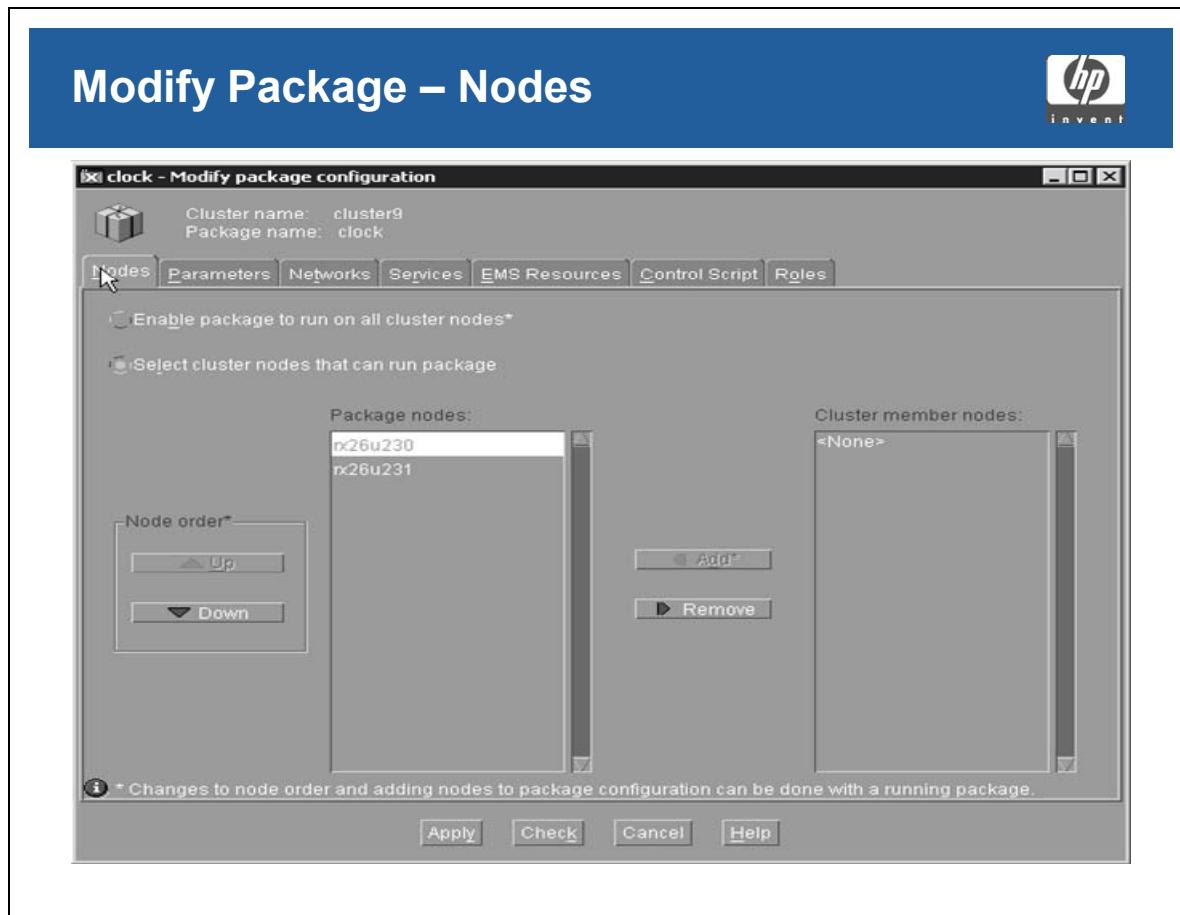
Additionally, the "Modify Package" selection allows the following:

- Changes to the order in which nodes will run the package
- Change initial setting for Auto Run
- Enable/Disable Local LAN Failover
- Changes to FAILOVER and FAILBACK Policies
- Changes to monitored subnets

Module 18  
**Serviceguard Manager**

- Addition/Deletion of package IP Addresses
- Addition/Modification/Deletion of services for the package
- Addition/Deletion/Deletion of EMS Resources
- Changes to the control script logfile
- Editing of the control script itself
- Access Control Policy changes

## 18-12. SLIDE: Modify Package – Nodes

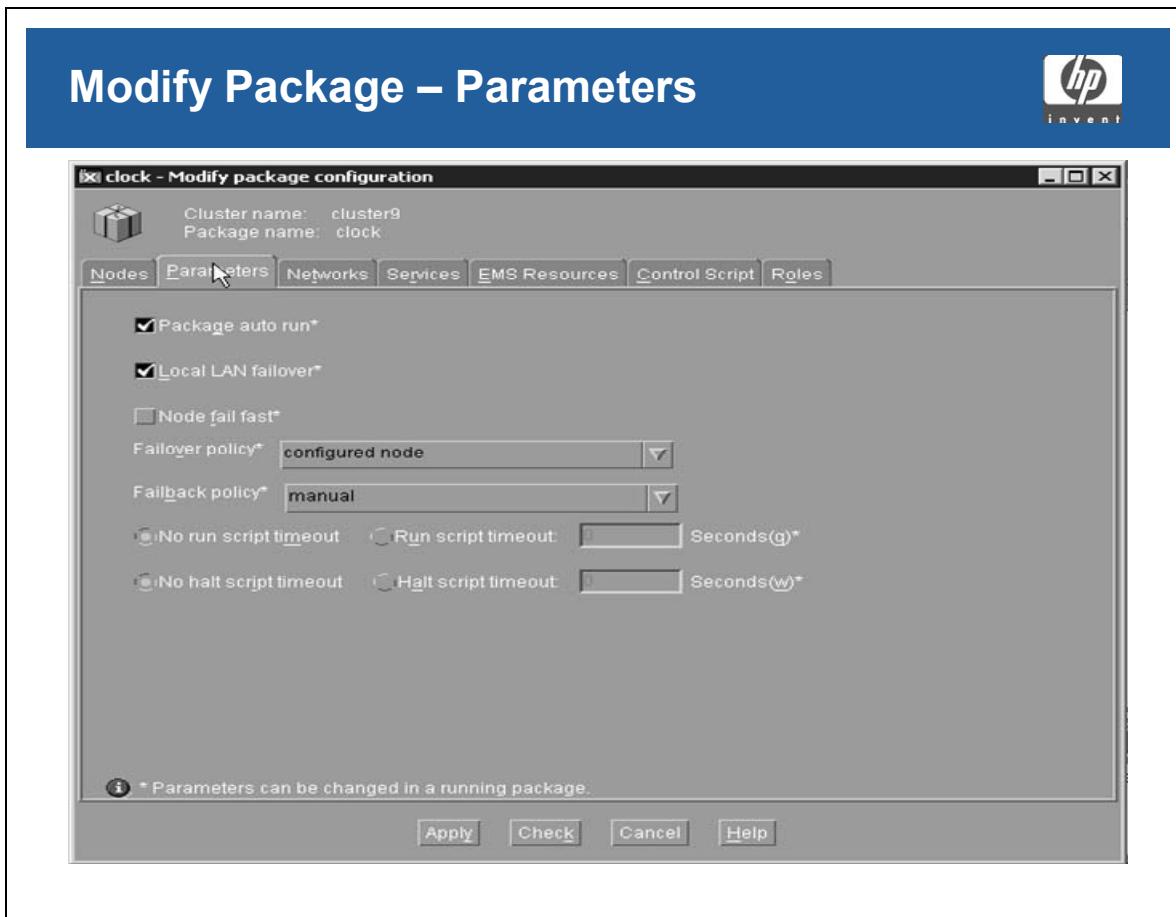


### Student Notes

The “Nodes” tab allows the following changes:

- Changes to the order of the nodes which will run the package
- Addition/Deletion of nodes that can run the package

## 18-13. SLIDE: Modify Package – Parameters

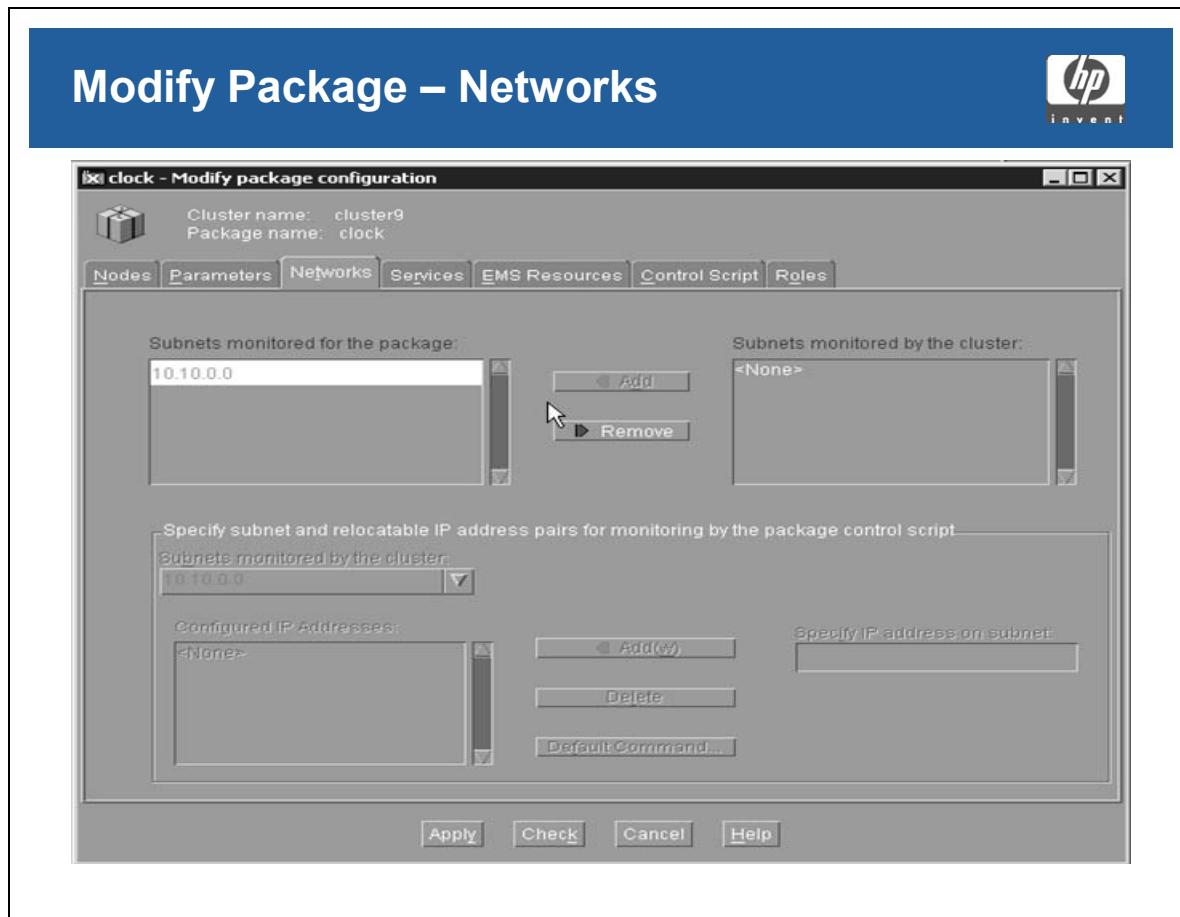


### Student Notes

The “Parameters” tab allows the following changes:

- Change initial setting for Auto Run
- Enable/Disable Local LAN Failover
- Changes to FAILOVER and FAILBACK Policies

## 18-14. SLIDE: Modify Package – Networks

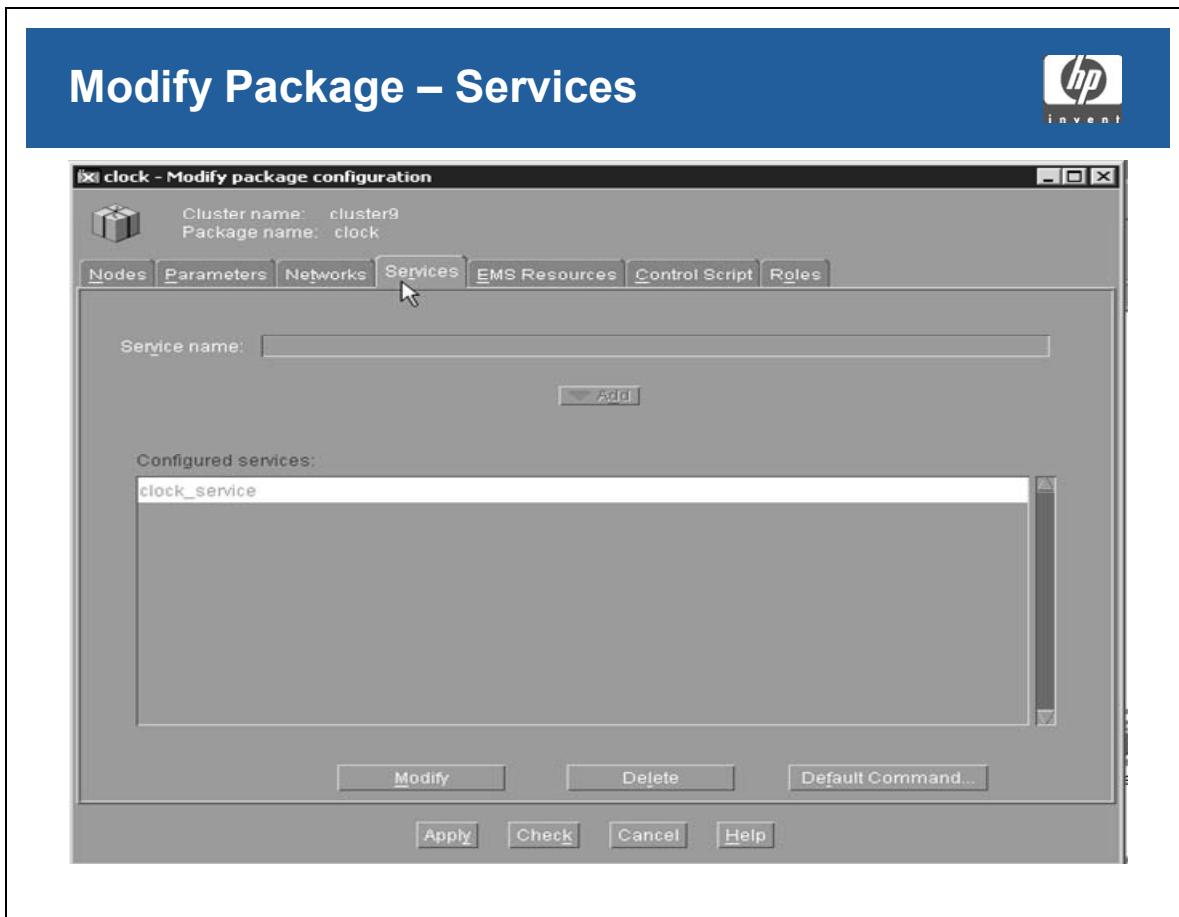


### Student Notes

The Networks tab allows the following changes:

- Changes to monitored subnets
- Addition/deletion of Package IP addresses

## 18-15. SLIDE: Modify Package – Services

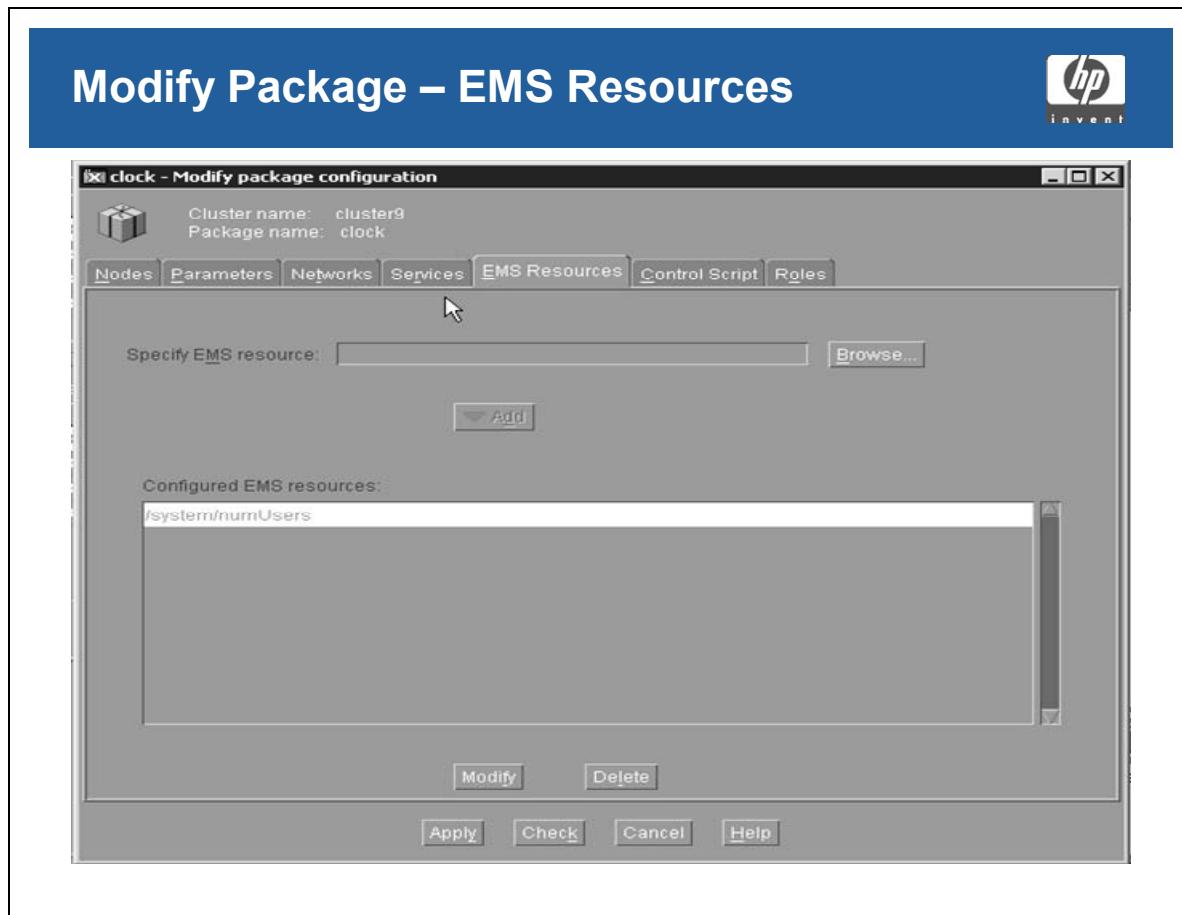


### Student Notes

The “Services” tab allows the following changes:

- Addition/Modification/Deletion of services for the package

## 18-16. SLIDE: Modify Package – EMS Resources

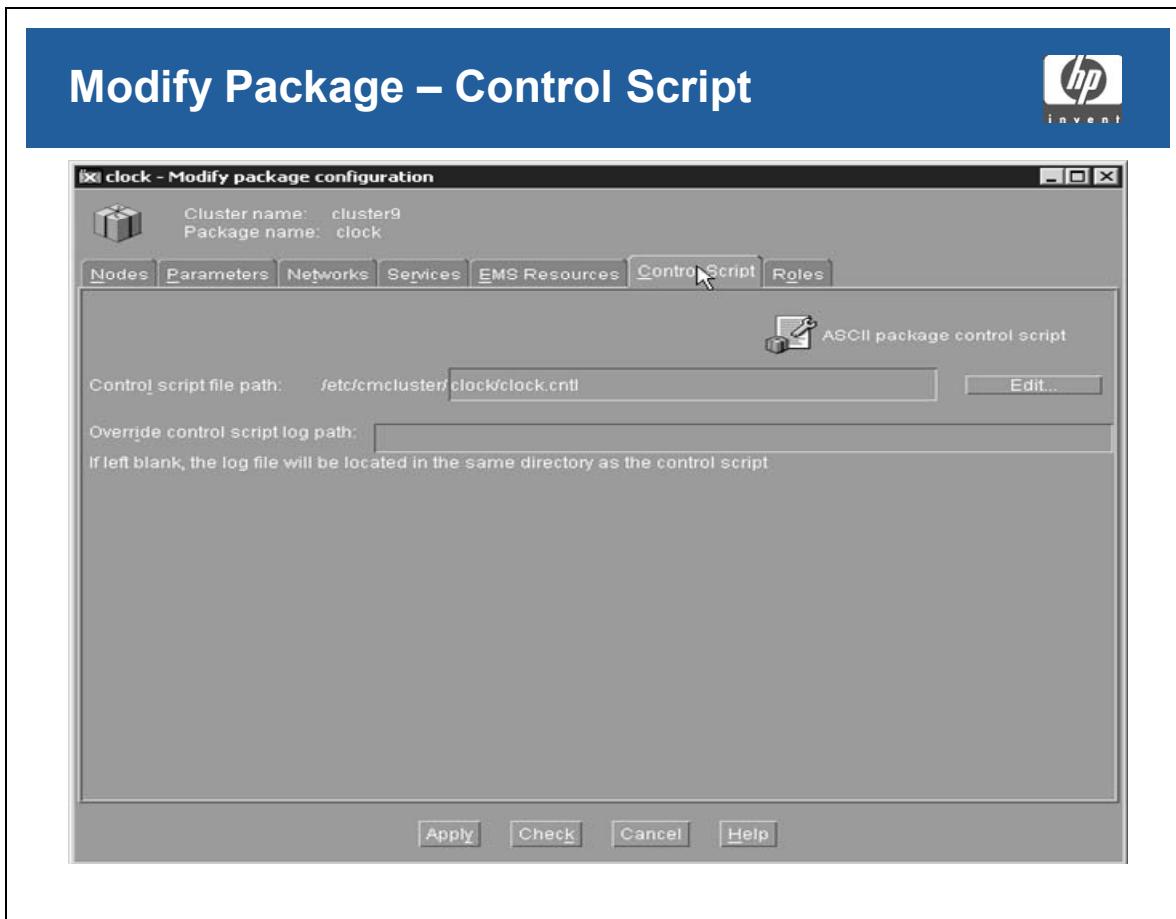


### Student Notes

The “EMS Resources” tab allows the following changes:

- Addition/Deletion/Modification of EMS Resources

## 18-17. SLIDE: Modify Package – Control Script

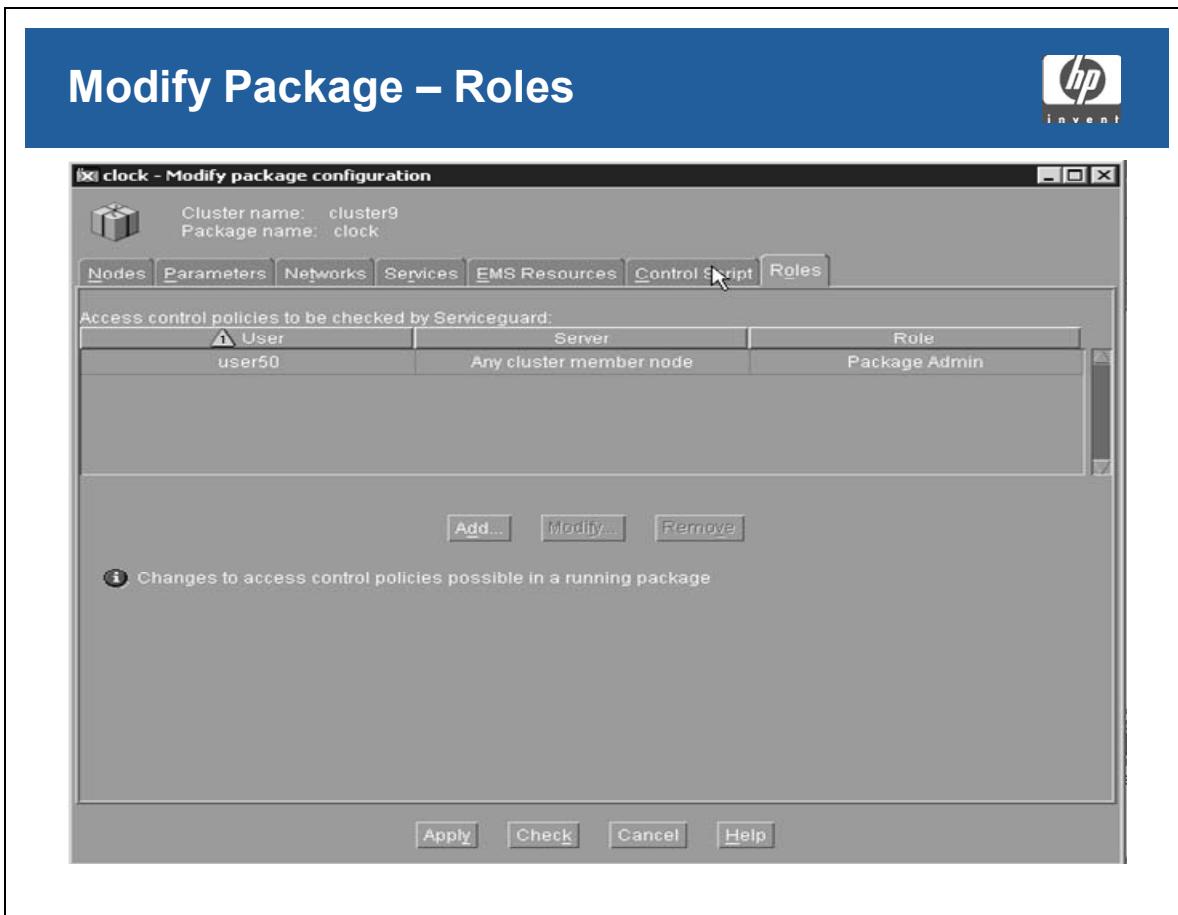


### Student Notes

The “Control Script” tab allows the following changes:

- Changes to the control script logfile path
- Editing of the control script itself

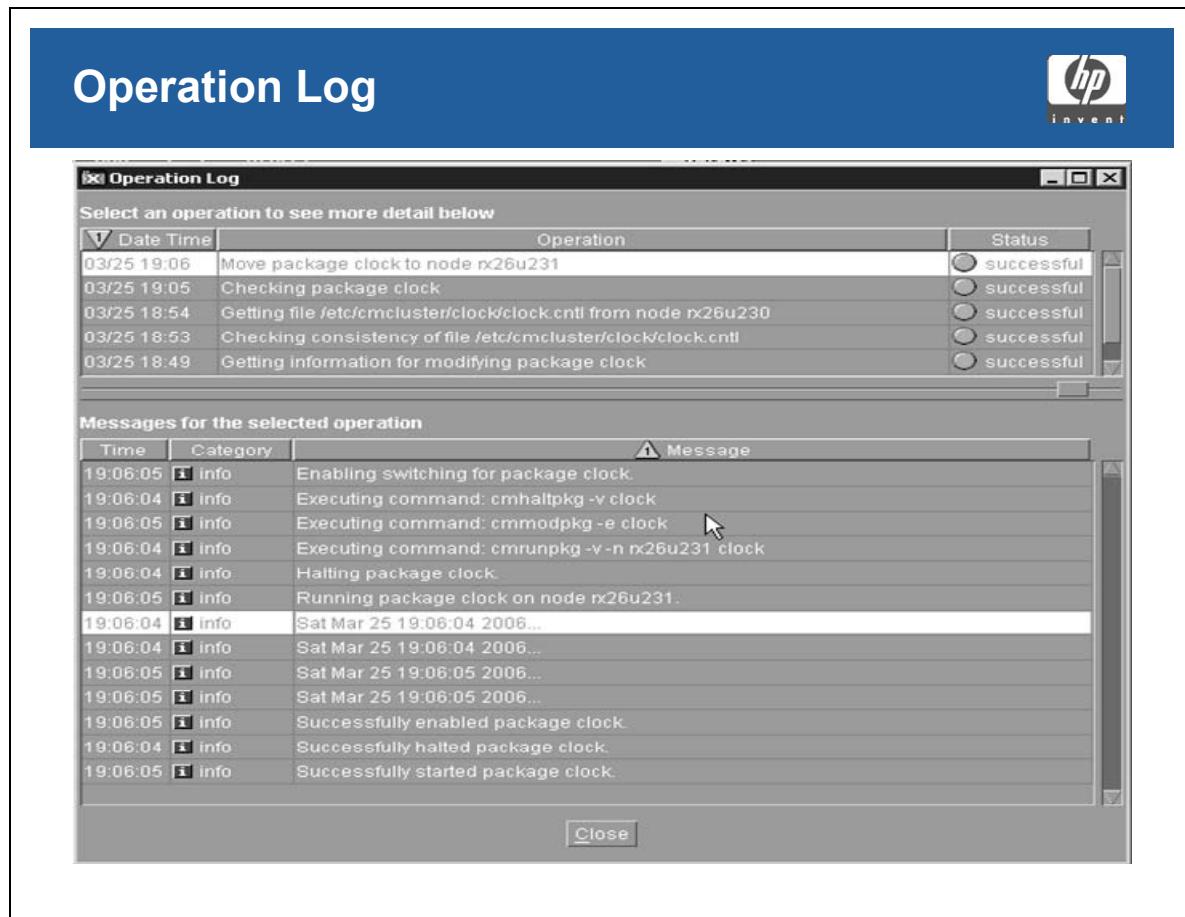
## 18-18. SLIDE: Modify Package – Roles



### Student Notes

The “Roles” tab allows changes to the Access Control Policy.

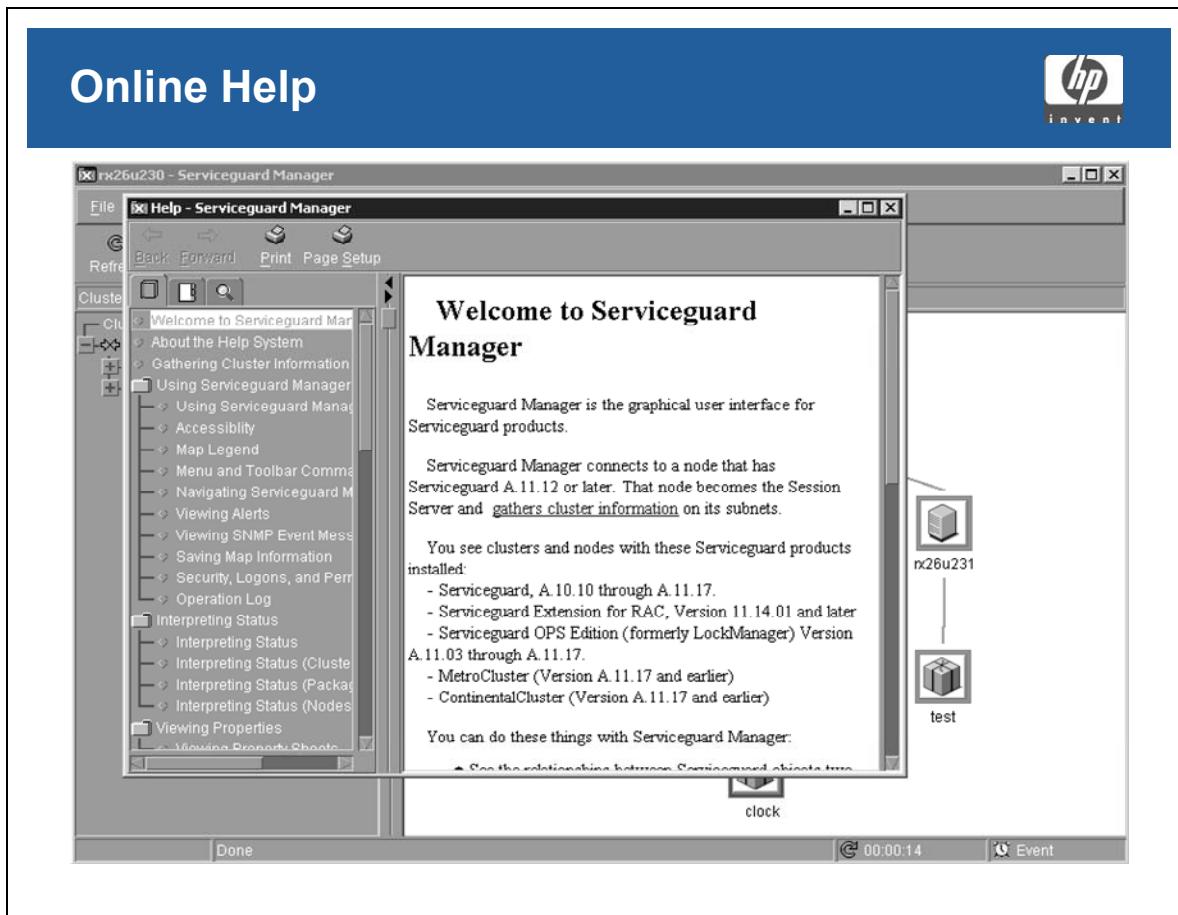
## 18-19. SLIDE: Operation Log



### Student Notes

Nearly every operation performed by Serviceguard Manager will launch an Operation Log window after completion. This window logs the actual commands that were executed to complete the task, a benefit to those operators who would like to understand better what is happening "behind the scenes".

## 18-20. SLIDE: Online Help



### Student Notes

#### Getting Help in Serviceguard Manager

Serviceguard Manager contains extensive help. The above screen shot shows only the first screen after clicking on the Serviceguard Manager Help button.

While you are working in Serviceguard Manager, you can open help on such topics as:

- Properties
- Map Legend
- Configuring Clusters
- Configuring Packages
- Troubleshooting

## **18-21. LAB: Exploring Serviceguard Manager**

### **Directions**

Complete the following tasks.

1. Start up Serviceguard Manager.

**Answer:**

```
# cd /opt/sgmgr/bin  
# ./sgmgr -c <your_cluster_name>
```

2. Log in to the Serviceguard Manager GUI.

**Answer:**

Server Name: node1  
User Name: root  
Password: <as specified by your instructor>

3. Explore the cluster shown in the map.

**Answer:**

On the left side of the screen, single-click on the ( + ) symbol next to the cluster name.

List the node names in the cluster and the package names in the cluster:

<b>Node Names in the Cluster</b>	<b>Packages Available in the Cluster</b>

4. On the right side of the screen, double click the cluster icon and note the same nodes and packages in the cluster – this time in graphical form.

Note: If the cluster or package is/are up, the icons have a green border. If the cluster or package is/are down, the icons have a red border.

5. Explore the properties for the cluster and nodes.

**Answer:**

Single-click on the icon for the cluster.

Right-click.

Select Properties of <cluster\_name>.

Select one or more of the property tabs shown.

After exploring several properties, close the properties window.

6. Using the Serviceguard Manager GUI, halt your cluster.

**Answer:**

Single-click on the cluster icon.

From the Actions pull-down menu, select Administration.

From the Administration pull-down, select Halt cluster <cluster\_name>.

If prompted with a Confirm Operation popup window, confirm this action by clicking OK.

Watch the Operations Log popup window to see a detailed outline of events.

Click Close to close the Operations Log popup window.

7. Using the Serviceguard Manager GUI, start your cluster.

**Answer:**

Single-click on the cluster icon.

From the Actions pull-down menu, select Administration.

From the Administration pull-down menu, select Run cluster <cluster\_name>.

At the Enable Network Probing popup window, click No.

If prompted with a Confirm Operation popup window, confirm this action by clicking OK.

Watch the Operations Log popup window to see a detailed outline of events.

Click Close to close the Operations Log popup window.

Module 18  
**Serviceguard Manager**

8. Using the Serviceguard Manager GUI, halt your test package, then observe the window in which the actual commands are logged.

**Answer:**

Instead of going to the Actions pull-down menu, let us try a slightly different procedure to accomplish the same task. Right click on the test package icon and select Halt package test.

If prompted with a Confirm Operation popup window, confirm this action by clicking OK.

Watch the Operations Log popup window to see a detailed outline of events.

Click Close to close the Operations Log popup window.

9. Now, start your test package on the other node.

**Answer:**

Right click on the test package icon and select Run package on node... Then select the other node.

If prompted with a Confirm Operation popup window, confirm this action by clicking OK.

Watch the Operations Log popup window to see a detailed outline of events.

Click Close to close the Operations Log popup window.

10. If there is a red "x" next to the test package icon, correct the problem.

**Answer:**

Right click on the test package icon.

Check the box next to "Package Switching".

If prompted with a Confirm Operation popup window, confirm this action by clicking OK.

Watch the Operations Log popup window to see a detailed outline of events.

Click Close to close the Operations Log popup window.

The red "x" next to the test package icon should now be gone. This indicates that all switches concerning this package's failover abilities have now been enabled.

11. Another feature of Serviceguard Manager is the ability to move a package. Currently, there is no command such as "cmmovepkg", but Serviceguard Manager can do it! To see this feature, move your test package back to its original node.

**Answer:**

Single-click on the test package icon.

From the Actions pull-down menu, select Administration.

From the Administration pull-down menu, click on Move Package to Node and select the node (to which the package will be moved) by clicking on the node name. If prompted to confirm this move, click OK.

Watch the Operations Log popup window to see a detailed outline of events.

Click Close to close the Operations Log popup window.

Explore the online Help.

**Answer:**

Select Help at the top of the Serviceguard Manager screen (the "?" icon).

Select one of the help topics.

Select another Help topic.

Can you print a Help topic?

Single-click the "X" in the upper right corner of the Serviceguard Manager "Help" window to close.

12. Test connecting to other clusters in the classroom.

**Answer:**

Select File from the menu.

Select New Session.

Enter server name, user name, and password for another cluster. In the Clusters to view section, select: View all clusters and unused nodes.

Select connect.

13. As a courtesy to the owner of this other cluster, ask their permission before continuing. As time and interest allow, investigate this new cluster using several different features of Serviceguard Manager.

14. Exit Serviceguard Manager.

**Answer:**

Click on File, then click on Exit.

## **18-22. LAB: Creating a Package with Serviceguard Manager**

### **Directions**

Complete the following tasks.

1. Start up Serviceguard Manager, if it is not already running.

#### **Answer:**

```
# cd /opt/sgmgr/bin  
# ./sgmgr -c <your_cluster_name>
```

2. If needed, log in to the Serviceguard Manager GUI.

#### **Answer:**

Server Name: node1  
User Name: root  
Password: <as specified by your instructor>

3. From the Actions pull down menu, select Configuration, and then click on Create Package.

#### **Answer:**

Click on Actions pull down menu.

Slide down to Configuration.

Click on Create Package.

4. If requested at the Authorization Dialog popup window, login as root with the appropriate password.

#### **Answer:**

Login: root

Password: < as specified by your instructor >

5. At the Operation Log popup window, you will see a message that says Getting Configuration for New Package ----- In progress. At this point, wait here for the next popup window.
6. In the Create Package Configuration popup window, ...
  - a. At this point, note that there are seven tabs across the top of the window. We will investigate each. Initially, we are first in the Nodes tab of the Create Package Configuration popup window.
  - b. With your mouse, highlight the generic name of the new package (New\_Package1), and type in xload. xload will be our next package that we configure.
  - c. Still in the Nodes tab, note that, by checking the appropriate radio button, we could decide to use only a subset of all the available cluster nodes for this new xload package. However, leave this at its default: Enable package to run on all cluster nodes.
  - d. Click on the Parameters tab. In this tab, note the different fields and how their values could be modified. For this lab, leave all values at their default.
  - e. Click on the Networks tab. In this tab, towards to right center, in the “Subnets monitored by the cluster” field, click on the 10-net subnet, and then, in the middle, click Add, to add this subnet to our package.
  - f. Note also, towards the bottom of this tab, that there is an area where we could configure IP addresses. Our xload package does not require an IP address, so leave these fields at their defaults.
  - g. Click on the Services tab. In this tab, we assign the Service\_Name of xload\_service. Type this service name into the specified field, then click Add. Note that, when you click Add, a new popup window appears. This window allows you to specify several values. For us, we need only specify the number of restarts **and** the Service command. Use 3 restarts, and the following Service command:  
`/usr/contrib/bin/X11/xload -display _____ -title _____`  
For your display option, launch a ReflectionX session and find the value of your DISPLAY variable. Use that value here. For title, use your first name. Then click OK.

- h. Click on the EMS Resources tab. In this tab, we could configure EMS resources. We will study EMS Resources in a later module. For now, leave this tab at its defaults.
- i. Click on the Control Script tab. Note that we configure LVM Volume Groups, File Systems, Customer Defined Functions, and/or vxVM Disk Groups here. Since we are not using any of these for our xload package, leave all fields at their defaults.

Module 18  
**Serviceguard Manager**

- j. Click on the `Roles` tab. In this tab, we could configure Access Control Policies for different users and different servers. Again leave all values at their defaults.
  - k. Click the `Apply` button at the lower center of your window. A popup appears that asks `Apply Configuration?` Click `OK` to apply the configuration.
  - l. This returns us to the `Operation Log` window, where we see a message that says `Checking & applying package xload --- in progress.`
  - m. Once the Operations Log window reports `Successful` ( instead of `in progress` ), click the close button to close the `Operations Log` window.
  - n. This returns us to the Serviceguard Manager main window. We should now see the `xload` package. At this point, single-click on the `xload` package icon, click on the `Actions` pull-down menu, click on `Administration`, and click on `Run package xload`. If prompted with a `Confirm Operation` popup window, click `OK` to begin the `xload` package.
  - o. This will bring an `Operations Log` popup window, where you can watch the progress as the `xload` package begins running. When you see the `Successfully started package xload` message, click `Close` to return you to the Serviceguard Manager main screen.
  - p. Notice that the `xload` package has a red "X" near the package icon. Right-click on the `xload` package icon, and click on `Properties of xload`. In the `xload Package Property Sheet` popup window, we can see that `Package Switching` is disabled. Click `Close` to close the popup window.
  - q. To enable `Package switching`, single-click on the `xload` icon, then click on the `Actions` pull down menu, then click on `Administration`. Place a check mark on `Package Switching`.
  - r. Another `Operation Log` popup window appears where we can watch the progress in real time. Wait for the `Successfully enabled package xload` message, and then click `OK` to close this window.
7. The `xload` package should now be running correctly with all switches enabled.
  8. Congratulations on configuring your first package using Serviceguard Manager!

---

## **Module 19 — Troubleshooting Scenarios and Final Review**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- Perform troubleshooting activities to resolve nine different Serviceguard configuration problems.
- Successfully complete 15 final review questions.

## **Some Important Notes**

There are two parts to this final module:

- "Troubleshooting Scenarios", and
- "Final review"

To accomplish both parts of this module, it is important that you accomplish the following:

- a. Before beginning the "Troubleshooting Scenarios" section, some minor editing is required. Go to the `/labs/mod_19/trbl` directory. Once there, change line 11 in the `trbl.sh` script to point to the disk you want to use for this section. This therefore means that you will need disk space available. If necessary, free up some disk space. Since we are now done with all packages previously configured, it is certainly ok to `cmdeleteconf` some packages and `vgexport` some volume groups in order to accomplish this.
- b. Also, the `trbl.sh` script will setup volume group `vg04`. Therefore, again, we will need disk space to be available. Again, it is certainly ok to `cmdeleteconf` some packages and `vgexport` some volume groups in order to accomplish this.
- c. Be sure that volume group `vg04` with minor number `0x040000` is not currently in use.
- d. Do not modify the `vg04` setting in the `trbl.sh` script. It works fine as is as long as you have an available disk.
- e. Ensure that there are no NFS clients still mounted. The setup script will remove the NFS package. Any remaining clients will therefore hang.
- f. Ensure the `"remsh"` functionality works between the nodes of your cluster. This implies that we need to create a `~root/.rhosts` file on each node in our cluster.
- g. The second part of this final module asks you to run `/mcsq/final/game`. You will note that some portions of this game are a little dated. None-the-less, it can still be a valuable learning experience.
- h. Good luck in this final module !!

***Please be sure to complete all of the above before continuing.***

## 19-1. SLIDE: Troubleshooting Scenarios

The screenshot shows a terminal window with a blue header bar containing the title "Troubleshooting Scenarios" and the HP Invent logo. Below the header, the command "# /labs/mod\_19/trbl/trbl.sh" is entered. A large, rounded rectangular box contains the program's output. The output starts with "TROUBLESHOOTING Scenarios" and a series of nine numbered troubleshooting items. At the bottom of the box, there is an instruction "Enter 1-9, (q)uit:".

```
# /labs/mod_19/trbl/trbl.sh
TROUBLESHOOTING Scenarios
*****
1. There exists a conflict within us
2. Missing the point ?
3. Busy, busy, busy

4. The import of this problem is crucial
5. What makes Johnny run ?
6. Fee, Fi, Foh, Fum

7. Care to comment ?
8. Thrown for a loop ?
9. What's in a name ?

*****
Enter 1-9, (q)uit:
```

### Student Notes

This module focuses on nine different troubleshooting scenarios that can exist in a Serviceguard environment. In this section of the module, troubleshoot as many of the above scenarios as possible as presented when you make a selection using the interactive simulation program. The program instructions will appear when you execute the startup script as described below.

The last section of this module contains a set of "final review" questions to test your knowledge of Serviceguard. The script will create a new volume group (**vg04**), logical volume (/dev/vg04/trbl), and mount point (trbl). It will also create a new package (/etc/cmcluster/pkg4).

To start the simulation program execute:

```
# cd /labs/mod_19/trbl
# ./trbl.sh
```

To exit one simulation and go on to another, type exit at the **MCSG>** prompt. To quit the troubleshooting scenario, at the Main menu, enter quit at the **MCSG>** prompt. Good luck!

## 19–2. SLIDE: A Conflict Exists within Us



### Student Notes

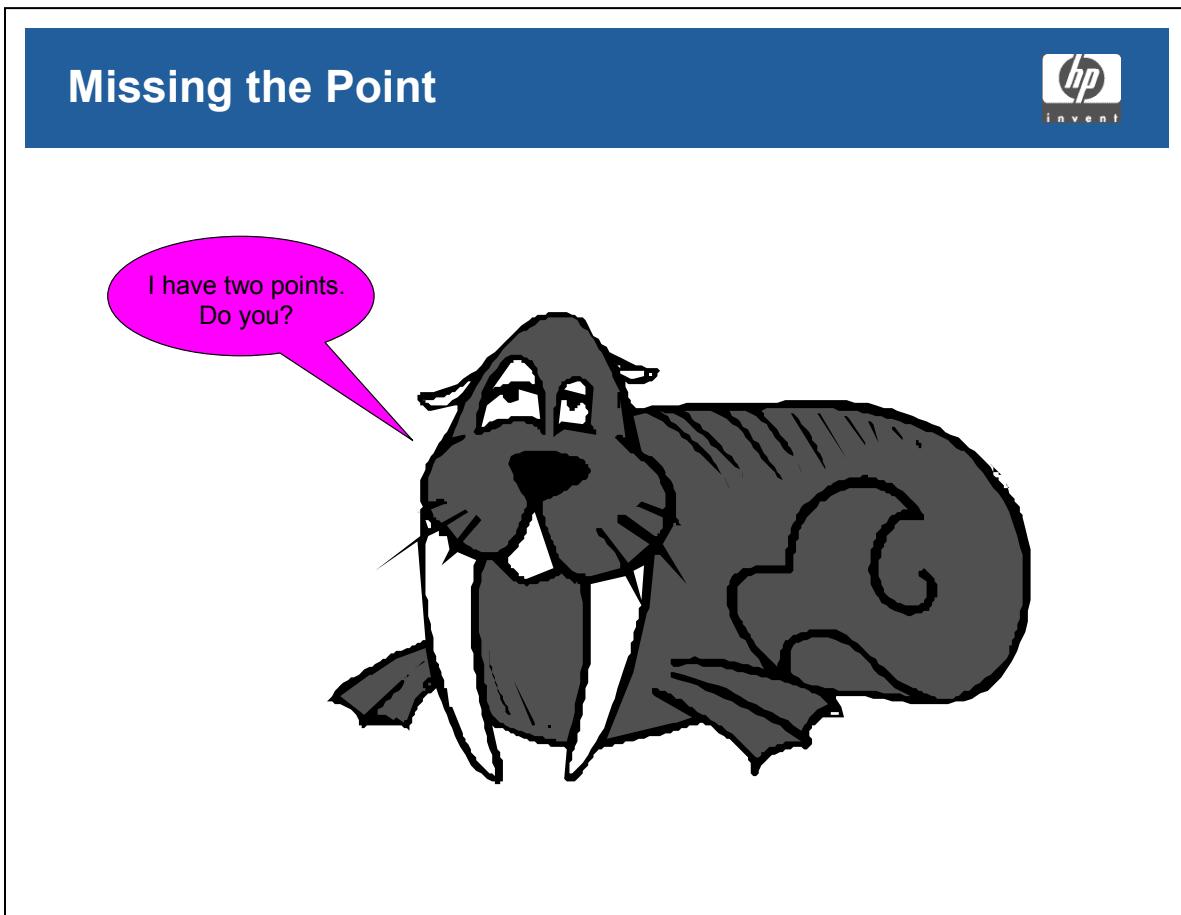
```
MCSG> cmrunc1
cmrunc1 : Waiting for cluster to form.....
cmrunc1 : Cluster successfully formed.
cmrunc1 : Check the syslog files on all nodes in the cluster
cmrunc1 : to verify that no warnings occurred during startup.

MCSG> cmrunpkg pkg4
cmrunpkg : Script failed with no restart: pkg4 should not be restarted.
Check the syslog on node r208c17 and pkg log files for more detailed information.

MCSG> cd /etc/cmcluster/pkg4

MCSG> tail *.log
#####
Node "r208c17": Starting package at Mon Mar 12 04:26:30 PST 2001
#####
Mar 12 04:26:30 - "r208c17": Activating volume group vg04 with exclusive option:
vgchange: Activation mode requested for the volume group "/dev/vg04" conflicts with
configured mode.
        ERROR: Function activate_volume_group
        ERROR: Failed to activate vg04
Mar 12 04:26:30 - Node "r208c17": Deactivating volume group vg04
vgchange: Volume group "vg04" has been successfully changed.
```

### 19-3. SLIDE: Missing the Point



#### Student Notes

```
MCSG> cmrunc1
cmrunc1 : Waiting for cluster to form....
cmrunc1 : Cluster successfully formed.
cmrunc1 : Check the syslog files on all nodes in the cluster
cmrunc1 : to verify that no warnings occurred during startup.

MCSG> cmrunpkg pkg4
cmrunpkg : Script failed with no restart: pkg4 should not be restarted.
Check the syslog on node r208c17 and pkg log files for more detailed information.

MCSG> cd /etc/cmcluster/pkg4

MCSG> tail *.log
Mar 12 05:27:19 - Node "r208c17": Checking filesystems:
/dev/vg04/trbl
file system is clean - log replay is not required
Mar 12 05:27:20 - Node "r208c17": Mounting /dev/vg04/trbl at /trbl
mount: /trbl: No such file or directory
```

\*\*\*\*\* Continued on next page \*\*\*\*\*

```
ERROR:  Function check_and_mount
ERROR:  Failed to mount /dev/vg04/trbl to /trbl
Mar 12 05:27:20 - Node "r208c17": Deactivating volume group vg04
Deactivated volume group in Exclusive Mode.
Volume group "vg04" has been successfully changed.
```

## 19-4. SLIDE: Busy, Busy, Busy



### Student Notes

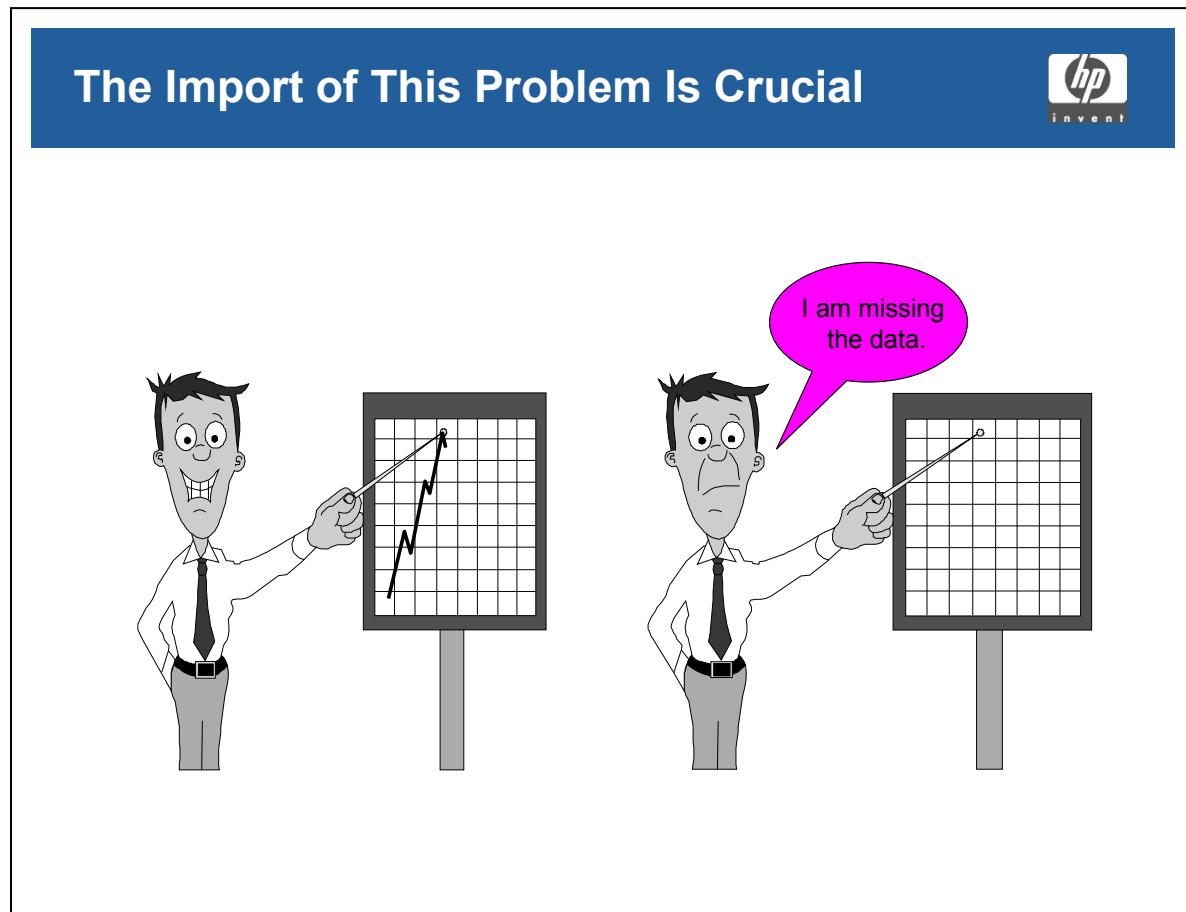
```
MCSG> cmrunc1
cmrunc1 : Waiting for cluster to form....
cmrunc1 : Cluster successfully formed.
cmrunc1 : Check the syslog files on all nodes in the cluster
cmrunc1 : to verify that no warnings occurred during startup.

MCSG> cmrunpkg pkg4
cmrunpkg : Script failed with no restart: pkg4 should not be restarted.
Check the syslog on node r208c17 and pkg log files for more detailed information.

MCSG> cd /etc/cmcluster/pkg4

MCSG> tail *.log
Mar 12 05:35:27.....Node "r208c17": Mounting /dev/vg04/trbl at /trbl
vxfs mount: /dev/vg04/trbl is already mounted, /trbl is busy,
or allowable number of mount points exceeded
ERROR: Function...check...and...mount
      ERROR: Failed to mount /dev/vg04/trbl to /trbl
Mar 12 05:35:28 - Node "r208c17": Deactivating volume group vg04
Deactivated volume group in Exclusive Mode.
Volume group "vg04" has been successfully changed.
```

## 19–5. SLIDE: The Import of This Problem Is Crucial



### Student Notes

```
MCSG> cmrunc1  
cmrunc1 : Waiting for cluster to form.....  
cmrunc1 : Cluster successfully formed.  
cmrunc1 : Check the syslog files on all nodes in the cluster  
cmrunc1 : to verify that no warnings occurred during startup.
```

```
MCSG> cmrunkpg pkg4  
cmrunkpg : Warning: Package pkg4 is already running.  
Check the syslog on node r208c17 and pkg log files for more detailed  
information
```

```
MCSG> cmhaltnode -f  
Disabling package switching to all nodes being halted.  
Warning: Do not modify or enable packages until the halt operation is  
completed
```

```
Halting cluster services on node r208c17  
cmhaltnode : Successfully halted all nodes specified.
```

Halt operation completed.

```
MCSG> cmrunkpkg -n node2 pkg4
cmrunkpkg : Script failed with no restart: pkg4 should not be restarted.
Check the syslog on node r208c18 and pkg log files for more detailed
information

MCSG> remsh node2 tail /etc/cmcluster/pkg4/pkg4.sh.log

##### Node "r208c18": Starting package at Mon Jul  7 20:45:49 PDT 2003#####
Jul  7 20:45:49 - "r208c18": Activating volume group vg04 with exclusive
option.

vgchange: Volume group "/dev/vg04" does not exist in the "/etc/lvmtab"
file.
ERROR: Function activate_volume_group
ERROR: Failed to activate vg04
Jul  7 20:45:49 - Node "r208c18": Deactivating volume group vg04
vgchange: Volume group "/dev/vg04" does not exist in the "/etc/lvmtab"
file.
ERROR: Function deactivate_volume_group
ERROR: Failed to deactivate vg04
```

## 19–6. SLIDE: What Makes Johnny Run?



### Student Notes

```
MCSG> cmrunc1  
cmrunc1 : Waiting for cluster to form.....  
cmrunc1 : Cluster successfully formed.  
cmrunc1 : Check the syslog files on all nodes in the cluster  
cmrunc1 : to verify that no warnings occurred during startup.
```

```
MCSG> cmrunpkg pkg4  
cmrunpkg : Warning: Package pkg4 is already running.  
Check the syslog on node r208c17 and pkg log files for more detailed  
information
```

```
MCSG> cmhaltnode -f  
Disabling package switching to all nodes being halted.  
Warning: Do not modify or enable packages until the halt operation is  
completed
```

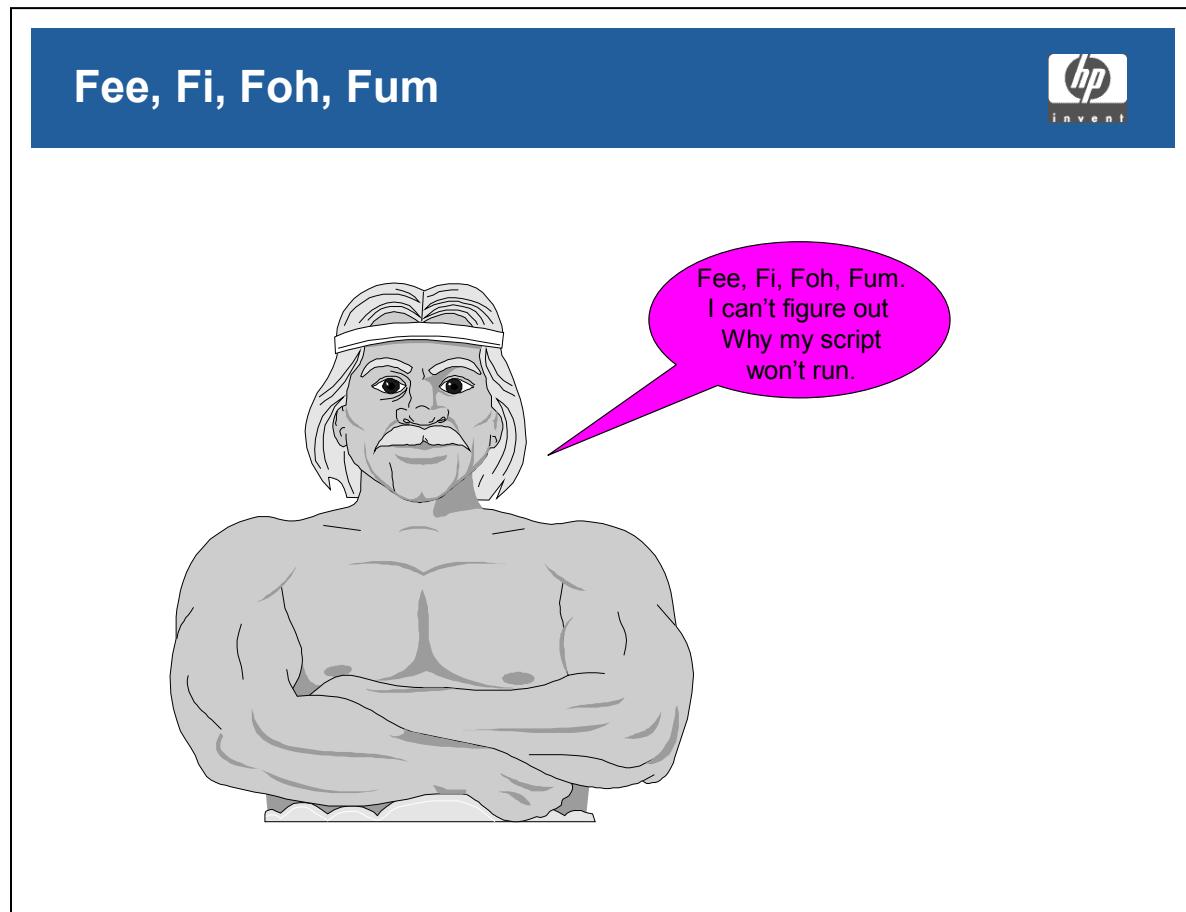
```
Halting cluster services on node r208c17  
..  
cmhaltnode : Successfully halted all nodes specified.  
Halt operation completed.
```

```
MCSG> cmrunpkg -n node2 pkg4
cmrunpkg : Unknown error returned from daemon: Error 0
Check the syslog on node r208c18 and pkg log files for more detailed
information
.

MCSG> remsh node2 tail /etc/cmcluster/pkg4/pkg4.sh.log

MCSG> remsh node2 tail /var/adm/syslog/syslog.log
Jul 7 20:49:23 r208c18 cmclld: 1 nodes have formed a new cluster, sequence
#2
Jul 7 20:49:23 r208c18 cmclld: The new active cluster membership is:
r208c18(id=2)
Jul 7 20:49:41 r208c18 cmclld: Request from node r208c18 to start package
pkg4 on node r208c18.
Jul 7 20:49:41 r208c18 cmclld: Executing '/etc/cmcluster/pkg4/pkg4.sh
start' for package pkg4, as service PKG*3073.
Jul 7 20:49:41 r208c18 cmsrvassistd[14428]: File
'/etc/cmcluster/pkg4/pkg4.sh' does not have execute permission.
Jul 7 20:49:41 r208c18 cmsrvassistd[14428]: Unable to start service
PKG*3073.
Jul 7 20:49:41 r208c18 cmclld: Package pkg4 run script failed.
Jul 7 20:49:41 r208c18 cmclld: Examine the file
/etc/cmcluster/pkg4/pkg4.sh.log for more details.
Jul 7 20:49:41 r208c18 cmclld: Disabled node r208c18 from running package
pkg4.
Jul 7 20:49:41 r208c18 cmclld: Unable to start package pkg4. Node r208c18
is not able to run it.
```

## 19-7. SLIDE: Fee, Fi, Foh, Fum



### Student Notes

```
MCSG> cmrunc1
cmrunc1 : Waiting for cluster to form....
cmrunc1 : Cluster successfully formed.
cmrunc1 : Check the syslog files on all nodes in the cluster
cmrunc1 : to verify that no warnings occurred during startup.

MCSG> cmrunpkg pkg4
cmrunpkg : Node is currently unable to run package pkg4.
Check the syslog on node r208c17 and pkg log files for more detailed information.

MCSG> cd /etc/cmcluster/pkg4

MCSG> tail *.log
/etc/cmcluster/pkg4/pkg4.sh[142]: Syntax error at line 150 : `}' is not expected.
```

## 19-8. SLIDE: Care to Comment?



### Student Notes

```
MCSG> cmrunc1
cmrunc1 : Waiting for cluster to form.....
cmrunc1 : Cluster successfully formed.
cmrunc1 : Check the syslog files on all nodes in the cluster
cmrunc1 : to verify that no warnings occurred during startup.

MCSG> cmrunpkg pkg4
cmrunpkg : Warning: Package pkg4 is already running.
Check the syslog on node r208c17 and pkg log files for more detailed
information

cmrunpkg : Completed successfully on all packages specified.

MCSG> cmhaltnode -f
Disabling package switching to all nodes being halted.
Warning: Do not modify or enable packages until the halt operation is
completed

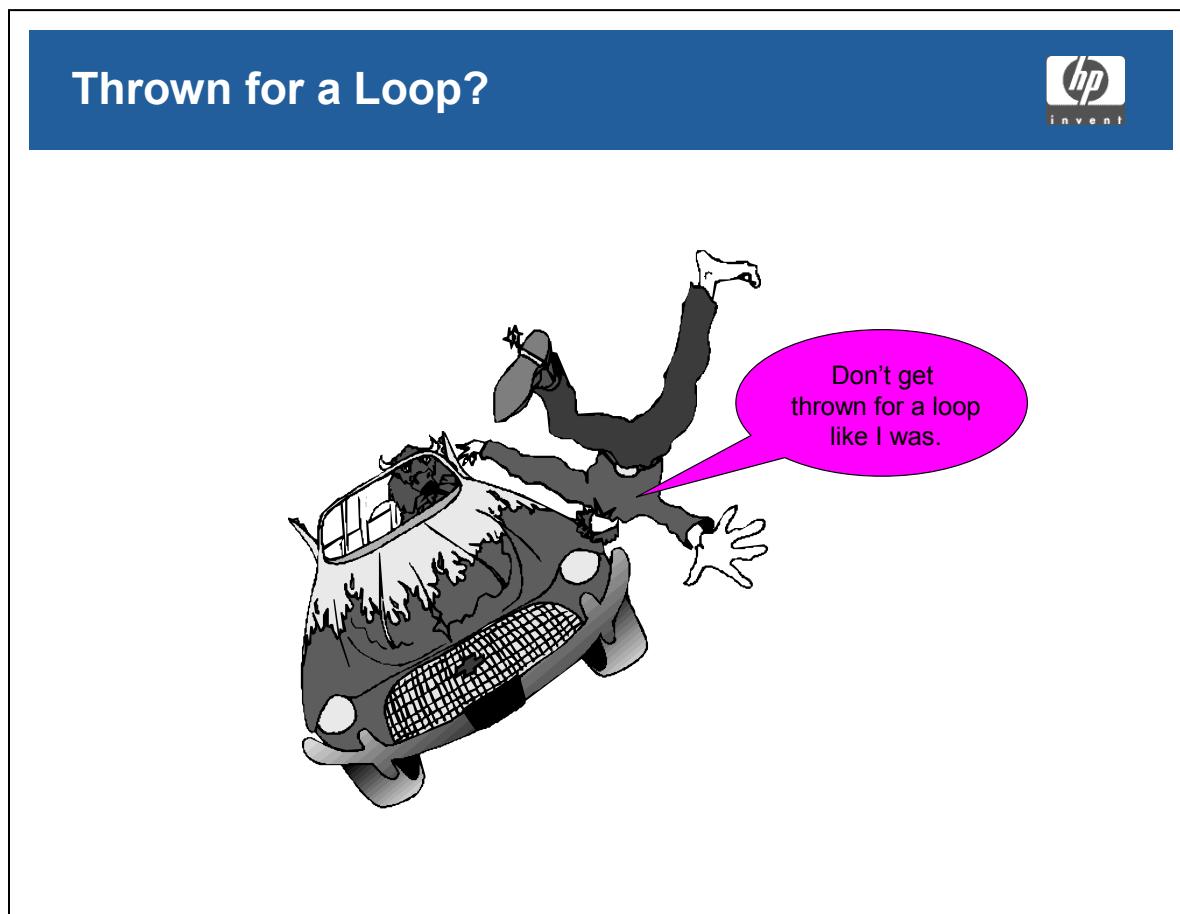
Halting Package pkg4
Halting cluster services on node r208c17
```

## Module 19

### Troubleshooting Scenarios and Final Review

```
cmhaltnode : Successfully halted all nodes specified.  
Halt operation completed.  
  
MCSG> cmrunpkg -n node2 pkg4  
cmrunpkg : Script failed with no restart: pkg4 should not be restarted.  
Check the syslog on node r208c18 and pkg log files for more detailed  
information  
  
MCSG> cd /etc/cmcluster/pkg4  
  
MCSG> tail *.log  
Volume group "vg04" has been successfully changed.  
  
#### Node "r208c17": Package start completed at Mon Jul 7 20:52:02 PDT  
2003 ####  
  
#### Node "r208c17": Halting package at Mon Jul 7 20:52:25 PDT 2003####  
Jul 7 20:52:25 - Node "r208c17": Deactivating volume group vg04  
Deactivated volume group in Exclusive Mode.  
Volume group "vg04" has been successfully changed.  
  
#### Node "r208c17": Package halt completed at Mon Jul 7 20:52:25 PDT  
2003 ####  
  
MCSG> tail /var/adm/syslog/syslog.log  
Jul 7 20:52:25 r208c17 cmclld: Enabled switching for package pkg4.  
Jul 7 20:52:25 r208c17 cmclld: Package pkg4 cannot run on this node because  
switching has been disabled for this node  
Jul 7 20:52:26 r208c17 cmtaped[20822]: cmtaped terminating. (ATS 1.14)  
Jul 7 20:52:27 r208c17 cmclld: Disabled switching for package pkg4.  
Jul 7 20:52:27 r208c17 cmclld: This node (r208c17) has ceased cluster  
activities  
Jul 7 20:52:27 r208c17 cmclld: Daemon exiting  
Jul 7 20:52:27 r208c17 cmsrvassistd[20819]: The cluster daemon aborted our  
connection.  
Jul 7 20:52:27 r208c17 cmsrvassistd[20819]: Lost connection with  
Serviceguard cluster daemon (cmclld): Software caused connection abort  
Jul 7 20:52:27 r208c17 cmclconfd[20741]: The cluster daemon aborted our  
connection.  
Jul 7 20:52:44 r208c17 CM-CMD[20859]: cmrunpkg -n r208c18 pkg4  
  
MCSG> remsh r208c18 tail /etc/cmcluster/pkg4/pkg4.sh.log  
  
#### Node "r208c18": Starting package at Mon Jul 7 20:52:47 PDT 2003 ####  
Jul 7 20:52:47 - "r208c18": Activating volume group vg04 with exclusive  
option.  
vgchange: Volume group "/dev/vg04" does not exist in the "/etc/lvmtab"  
file.  
ERROR: Function activate_volume_group  
ERROR: Failed to activate vg04  
Jul 7 20:52:47 - Node "r208c18": Deactivating volume group vg04  
vgchange: Volume group "/dev/vg04" does not exist in the "/etc/lvmtab"  
file.  
ERROR: Function deactivate_volume_group  
ERROR: Failed to deactivate vg04
```

## 19-9. SLIDE: Thrown for a Loop?



### Student Notes

```
MCSG> cmrunc1 -v
cmrunc1 : Waiting for cluster to form.....
cmrunc1 : Cluster successfully formed.
cmrunc1 : Check the syslog files on all nodes in the cluster
cmrunc1 : to verify that no warnings occurred during startup.

MCSG> cmrunpkg -v pkg4
cmrunpkg : Script failed with no restart: pkg4 should not be restarted.
Check the syslog on node r208c17 and pkg log files for more detailed
information

MCSG> tail /var/adm/syslog/syslog.log
Jul  3 19:37:40 r208c17 CM-CMD[12547]: cmrunpkg pkg4
Jul  3 19:37:40 r208c17 cmclld: Request from node r208c17 to start package
pkg4 on node r208c17.
Jul  3 19:37:40 r208c17 cmclld: Executing '/etc/cmcluster/pkg4/pkg4.sh
start' for package pkg4, as service PKG*50689.
Jul  3 19:37:42 r208c17 LVM[12558]: vgchange -a e vg04
Jul  3 19:37:43 r208c17 LVM[12596]: vgchange -a n vg04
```

## Module 19

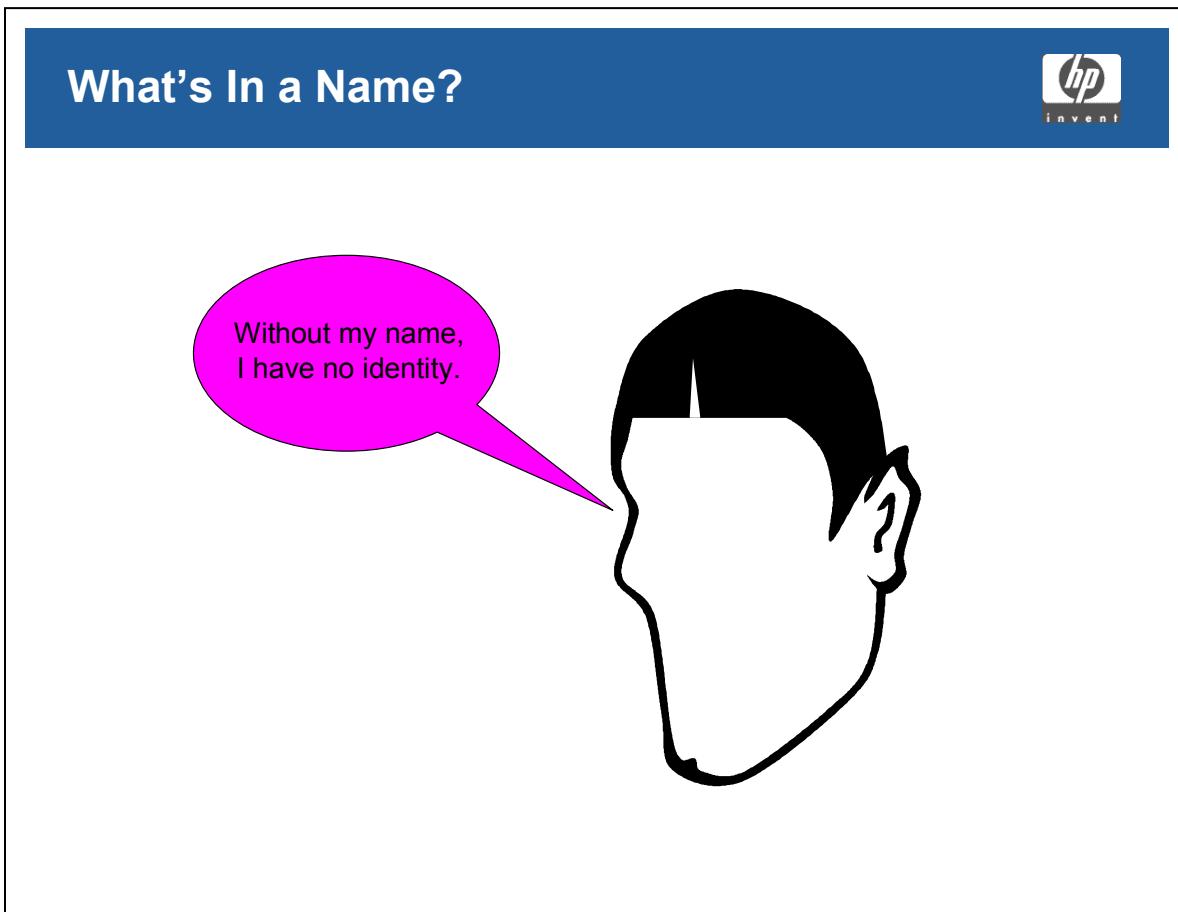
### Troubleshooting Scenarios and Final Review

```
Jul  3 19:37:43 r208c17 cmclld: Service PKG*50689 terminated due to an
exit(1).
Jul  3 19:37:43 r208c17 cmclld: Package pkg4 run script exited with
NO_RESTART.
Jul  3 19:37:43 r208c17 cmclld: Examine the file
/etc/cmcluster/pkg4/pkg4.sh.log for more details.
Jul  3 19:37:43 r208c17 cmclld: Switching disabled on package pkg4.
Jul  3 19:37:43 r208c17 cmclld: Unable to start package pkg4. Node r208c17
is not able to run it.
```

```
MCSG> cd /etc/cmcluster/pkg4
```

```
MCSG> tail *.log
/dev/vg04/trbl
file system is clean - log replay is not required
Jul  7 20:55:47 - Node "r208c17": Mounting /dev/vg04/trbl at /trbl
vxfs mount: /dev/vg04/trbl is already mounted, /trbl is busy,
        or allowable number of mount points exceeded
        ERROR: Function check_and_mount
        ERROR: Failed to mount /dev/vg04/trbl to /trbl
Jul  7 20:55:47 - Node "r208c17": Deactivating volume group vg04
Deactivated volume group in Exclusive Mode.
Volume group "vg04" has been successfully changed.
```

## 19-10. SLIDE: What's in a Name?



### Student Notes

```
MCSG> cmrunc1
cmrunc1 : Waiting for cluster to form....
cmrunc1 : Cluster successfully formed.
cmrunc1 : Check the syslog files on all nodes in the cluster
cmrunc1 : to verify that no warnings occurred during startup.

MCSG> cmrunpkg pkg4
cmrunpkg : Script failed with no restart: pkg4 should not be restarted.
Check the syslog on node r208c17 and pkg log files for more detailed information.

MCSG> cd /etc/cmcluster/pkg4

MCSG> tail *.log
/dev/vg04/trbl
file system is clean - log replay is not required
Jul  7 21:00:09 - Node "r208c17": Mounting /dev/vg04/trbl at /trbl
vxfs mount: /dev/vg04/trbl is already mounted, /trbl is busy, or allowable number
of mount points exceeded
      ERROR: Function check_and_mount
      ERROR: Failed to mount /dev/vg04/trbl to /trbl
Jul  7 21:00:09 - Node "r208c17": Deactivating volume group vg04
Deactivated volume group in Exclusive Mode.
```

## 19-11. SLIDE: Final Review

# Final Review



FUN with Serviceguard

---

... A review of Serviceguard concepts & commands ...  
for

The clear of mind .....  
The sturdy of will .....  
The adventurous of heart ....

Press RETURN to continue, (q)uit:

### Student Notes

To start the review, enter the following at the command prompt:

```
# cd /labs/mod_19/final
```

Once in this directory, you will see two versions of the game program: the first is game\_parisc ( for use in those classes using PA-RISC equipment ), while the second is game\_integrity ( for use in those classes using Integrity equipment ). If you are not sure what kind of equipment you are using, ask your instructor.

```
# ./game_parisc
```

or

```
# ./game_integrity
```

This "question and answer" game serves as a review of cluster and package concepts and commands encountered during this class.

Multiple-choice questions are randomly chosen from a large "pool" of available questions; thus, if you replay the game, you may see questions not yet encountered. Your score is increased by one for each question answered correctly; however, it decreases by one for each question missed.

To "complete" the game, you must tally 15 points. If all possible questions from the pool have been presented, and you have not yet reached 15 points, the game will cease at that point.

You may quit the game at any time by answering '**q**' to the current question. Partial games are not saved, and thus cannot be resumed.

## 19-12. LAB Solution: Troubleshooting Scenarios

### Scenario #1 There exists a conflict within us

- Hint: The package (pkg4) should be running on its primary node.  
It should activate vg04 and mount /dev/vg04/trbl on /trbl.
- Problem: vg04 fails to activate.  
The exclusive activation mode bit for vg04 is not set.
- Solution: Start the cluster.  
Reset the activation mode bit. (vgchange -c y vg04)  
Run the package.

### Scenario #2 Missing the point

- Hint: The package (pkg4) should be running on its primary node.  
It should mount /dev/vg04/trbl on /trbl.
- Problem: The mount fails  
The mount point (/trbl) is missing.
- Solution: Create the mount point (mkdir /trbl)  
Run the package.

### Scenario #3 Busy, busy, busy

- Hint: The package (pkg4) should be running on its primary node.  
It should activate vg04 and mount /dev/vg04/trbl on /trbl.
- Problem: The mount fails.  
The mount point (/trbl) is busy.
- Solution: Identify processes using the mount point (fuser /trbl).  
Kill the processes.  
Run the package.

### Scenario #4 The import of this problem is crucial!

Hint:      The package (pkg4) should be running on its primary node.  
              It should activate vg04 and mount /dev/vg04/trbl on /trbl.  
              Halt the primary node (cmhltnode -f)  
              The package (pkg4) should switch to the other node.

Problem:     The package does not switch.  
              vg04 fails to activate.  
              vg04 is not defined on the other node.

Solution:     Import the vg04 definition on the other node.  

```
mkdir /dev/vg04
mknod /dev/vg04/group c 64 0x040000
vgimport -m /tmp/vg04.map vg04 /dev/dsk/cXtYdZ
```

Run the package on the adaptive node.

### Scenario #5 What makes Johnny run?

Hint:      The package (pkg4) should be running on its primary node.  
              It should activate vg04 and mount /dev/vg04/trbl on /trbl.  
  
              Halt the primary node (cmhltnode -f)  
              The package (pkg4) should switch to the other node.  
              Look in syslog for helpful information.

Problem:     The package does not switch.  
              The package control script (pkg4.sh) is not executable.

Solution:     Make the package control script executable (chmod 700 pkg4.sh)  
              Run the package on the adoptive node.

### Scenario #6 Fee, Fi, Foh, Fum

Hint:      The package (pkg4) should be running on its primary node.  
              It should activate vg04 and mount /dev/vg04/trbl on /trbl.

Problem:     The customer\_defined\_run\_cmds function contains a syntax error.  
              The if statement is missing a fi

Solution:     Add the needed fi to the customer\_defined\_run\_cmds function.  
              Run the package.

### Scenario #7 Care to comment?

Hint: The package (pkg4) should be running on its primary node.  
It should activate vg04 and mount /dev/vg04/trbl on /trbl.

Halt the primary node ( cmhaltnode -f ).  
The package (pkg4) should switch to the other node.  
It should activate vg04 and mount /dev/vg04/trbl on /trbl.

Problem: The package switches but does not mount the lvol.  
The lvol definition is commented out in the control script.

Solution: Halt the package.  
Uncomment the lvol definition in the control script (pkg4.sh)  
Run the package on the adoptive node.

### Scenario #8 Thrown for a loop

Hint: The package (pkg4) should be running on its primary node.  
It should activate vg04 and mount /dev/vg04/trbl on /trbl.

It should run a service named "servmon".  
Check the status of the package after a 10-15 second wait.

Look in syslog.log for helpful information.

Problem: The package starts but halts shortly thereafter.  
The service program exit causes the package to switch/halt.

Solution: Add an infinite loop to the servmon code to prevent the exit.  
Run the package.

### Scenario #9 What's in a name?

Hint: The package (pkg4) should be running on its primary node.  
It should activate vg04 and mount /dev/vg04/trbl on /trbl.  
It should run a customer-defined program.

Problem: Error in the customer\_defined\_run\_cmds function.  
The program name should be: /labs/mod\_17/trbl/start\_daemons

Solution: Fix the program name in the customer function.  
Run the package.

---

# Appendix A — Serviceguard Command Summary

## Serviceguard Commands

/usr/sbin/cmapplyconf	/usr/sbin/cmhaltserver	/usr/sbin/cmrunnode
/usr/sbin/cmcheckconf	/usr/sbin/cmmakepkg	/usr/sbin/cmrumpkg
/usr/sbin/cmdeleteconf	/usr/sbin/cmmigrate	/usr/sbin/cmrunserver
/usr/sbin/cmgetconf	/usr/sbin/cmmodnet	/usr/sbin/cmscancl
/usr/sbin/cmhaltserv	/usr/sbin/cmmodpkg	/usr/sbin/cmviewcl
/usr/sbin/cmhaltnode	/usr/sbin/cmquerycl	/usr/sbin/cmviewconf
/usr/sbin/cmhaltpkg	/usr/sbin/cmruncl	

## Serviceguard Command Summary

cmapplyconf	Verify and apply Serviceguard cluster configuration and package configuration files
cmcheckconf	Check cluster configuration and/or package configuration files
cmdeleteconf	Delete either the cluster or the package configuration
cmgetconf	Get cluster or package configuration information
cmhaltserv	Halt/stop a high availability cluster
cmhaltnode	Halt/stop a node in a high availability cluster
cmhaltpkg	Halt/stop a high availability package
cmhaltserv*	Halt/stop a service from the high availability package halt script
cmmakepkg	Create a high availability package template file
cmmigrate	Migrate SwitchOver/UX to a Serviceguard cluster configuration file
cmmodnet*	Add or remove an IP address from a high availability cluster network
cmmodpkg	Enable or disable switching attributes for a high availability package
cmquerycl	Query cluster or node configuration information, create cluster configuration
cmruncl	Run/start a high availability cluster
cmrunnode	Run a node in an existing high availability cluster
cmrumpkg	Run/start a high availability package
cmrunserver*	Run/start a service from the high availability package run script
cmscanc1	Gather system configuration information from nodes with Serviceguard installed.
cmviewcl	View information about a high availability cluster
cmviewconf	View Serviceguard cluster configuration information

\* These commands are normally used within Serviceguard package scripts, not from the command line.

## Contributed Commands

/usr/contrib/bin/cmsetlog	<ul style="list-style-type: none"><li>• Adjust the log level(detail) of daemons</li></ul>
/usr/contrib/bin/cmsetsafety	<ul style="list-style-type: none"><li>• Disable/enable the kernel safety timer</li></ul>

**Appendix A**  
**Serviceguard Command Summary**

```
***** W A R N I N G *****  
*  
* This is a special support tool supplied for debugging purposes *  
* only. It should only be used by trained and authorized HP personnel. *  
* Unauthorized or incorrect use of this tool can result in a system *  
* failure and possible data loss or corruption. Please contact your *  
* HP support representative. *  
*****
```

---

## **Appendix B — Managing Serviceguard with HP OpenView**

---

## **B-1. SLIDE: OpenView Products**

### **Openview Products**



- Generic SNMP Traps
- Network Node Manager
- Openview Operations for Unix and Windows
- Other Openview Products

#### **Student Note**

In this module we will examine the ways that Serviceguard can be managed effectively with Openview products and other frameworks.

SNMP Management is a technique that can be used if you are only using network management tools or 3<sup>rd</sup> party frameworks that communicate primarily via SNMP. Serviceguard provides a SNMP subagent that can send alarms when cluster events happen and can be used to poll for cluster and package status.

Network Node Manager is the premier network management platform in use. It is a very mature platform for effective network and system discovery and management. It can receive alarms via snmp traps and poll cluster data via snmp.

Openview Operations is for system and application management since it employs agent technology to monitor and collect performance data on the Operating System and applications on a host.

We will then look at some additional Openview products that can be used to manage clusters and the applications on the systems. In addition, many of the Openview products which run on HP-UX can be configured to be made highly available as a cluster package.

## B-2. SLIDE: SNMP Notification

# SNMP Notification

The HP Invent logo, featuring the lowercase letters "hp" in a stylized font inside a square, with the word "invent" in a smaller sans-serif font below it.

- **Serviceguard Notification**
  - Log files (syslog.log)
  - SNMP Traps
- **SNMP Traps**
  - Sent by `cmsnmpagt` ( Disabled by Default )
    - `/etc/rc.config.d/cmsnmpagt`  
`AUTOSTART_CMSNMPD=1`
    - By all nodes every time cluster event happens
  - All trap destinations listed in  
`/etc/snmpd.conf` -> `/etc/SnmpAgent.d/snmpd.conf`

## **Student Note**

By now you have seen the type of messages that Serviceguard puts in log files. This is the primary method by which you should be able to track cluster events.

The cluster events generate an SNMP trap which is a network packet that is sent out by each node in your cluster. An SNMP trap is a standard network management protocol that is used by all of the major network management products and devices.

To allow your system to send SNMP traps you need to enable this feature in the file /etc/rc.config.d/cmsnmpagt

```
# AUTOSTART_CMSNMPD: If set to 1, the CM cluster SNMP subagent  
# will be started automatically when the  
# system boots. Set to 1 if you will be using  
# ClusterView to monitor your Highly Available  
# clusters through OpenView. Setting AUTOSTART_CMCLD  
# to 1 in the /etc/rc.config.d/cmcluster file will  
# cause the node to attempt to join its CM cluster  
# automatically when the system boots.  
  
# If set to 0, the CM cluster SNMP subagent  
# will not be started.
```

## **Appendix B**

### **Managing Serviceguard with HP OpenView**

```
#  
AUTOSTART_CMSNMPD=1
```

The agent that sends the traps will be started with the next reboot. If you wish to restart the agent without rebooting execute the following command.

```
# /sbin/init.d/cmsnmpagt start
```

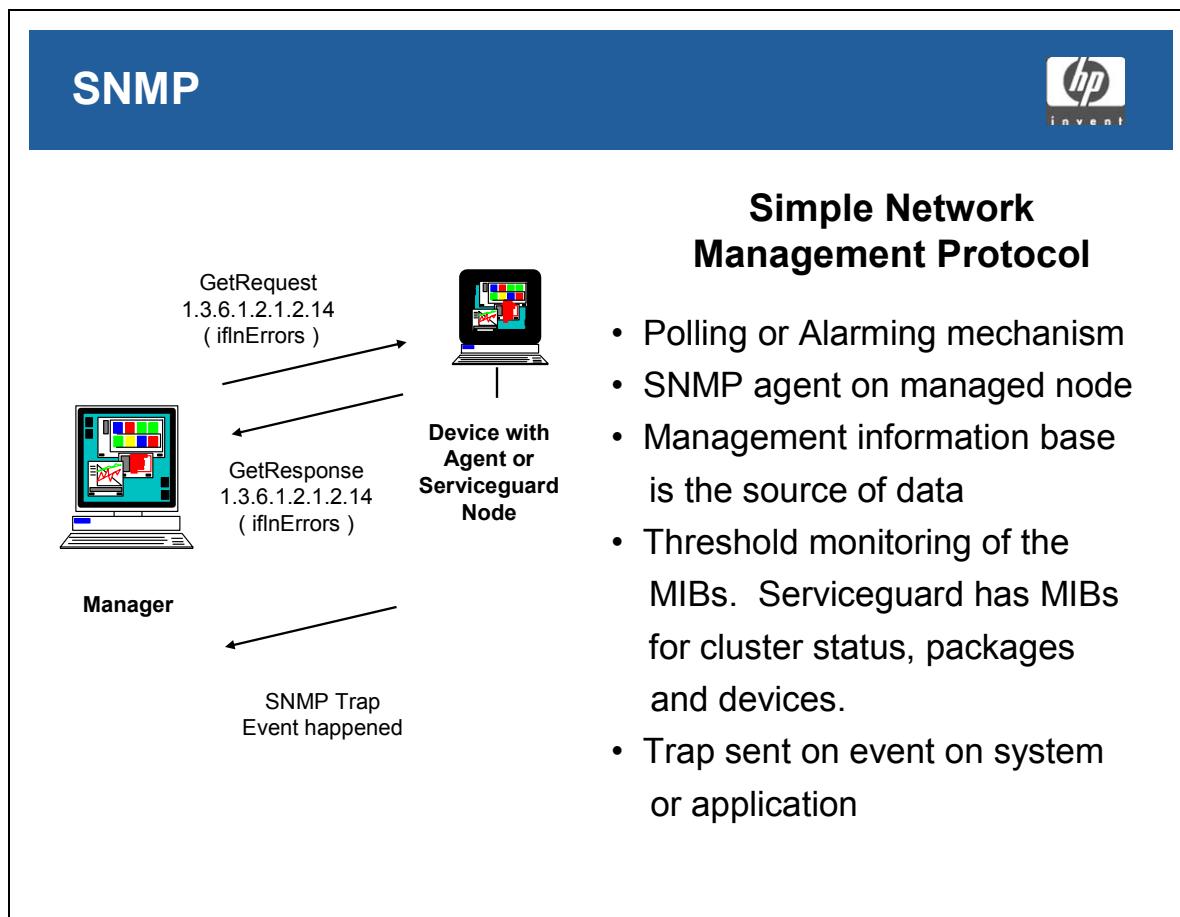
The agent sends traps to all "trap-dest" listed in the file /etc/snmpd. If your network management station is not listed in this file automatically, you must manually add it. If you manually change this file you must restart the SNMP agent on the system as well as the Cluster subagent. A reboot will do this, or you can execute

```
# ps -ef | grep snmpd  
# kill <PID>  
# cd /etc/snmpd  
# /sbin/rc.init.d/cmsnmpagt stop  
# /sbin/rc.init.d/cmsnmpagt start
```

The /etc/snmpd.conf file looks like:

```
#####  
#  
# snmpd.conf - HP / SNMP Research EMANATE SNMP Agent Configuration File  
#  
#####  
#  
# The EMANATE SNMP Agent included with HP OpenView Network Node Manager  
# uses two configuration files to initialize operational settings on agent  
# startup. They are:  
#  
#      snmpd.conf      Provides an HP OpenView SNMP Agent compatible  
#      (this file)       configuration file format for the EMANATE SNMP  
#                      Agent. The configuration values in this file  
#                      override and/or supplement the configuration  
#                      in snmpd.cnf.  
#  
#####  
# get-community-name:    public  
#set-community-name:    # enter community name  
#contact:              # enter contact person for agent  
#location:             # enter location of agent  
#max-trap-dest:        # enter max no. of trap-dest entries to be maintained.  
  
#trap-dest:             # enter trap destination  
trap-dest: 156.153.206.68  
trap-dest: 156.153.206.67
```

## B-3. SLIDE: SNMP



### Student Note

- SNMP is a standards based protocol that NNM uses to get data from the device or program to the management station.
- There are three major components to this action: Manager, Agent and the MIB.
- In addition to the SNMP agent on the managed device being the initiator, you can configure Network Node manager to manage any MIB object on ANY/ALL devices on your network and when a metric on that device exceeds a threshold, that SNMP agent will send an alarm to NNM.
- SNMP is a polling mechanism or can be used to send alarms (traps) when something happens.

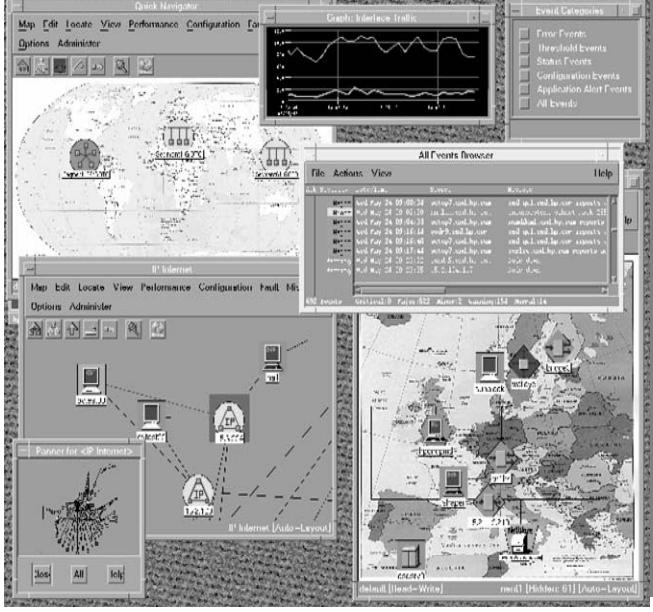
## B-4. SLIDE: What Is Network Node Manager?

# Network Node Manager



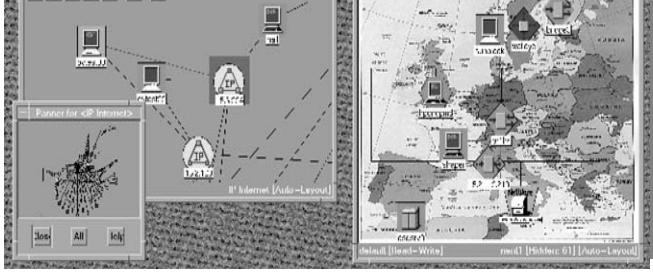
### Device Management

- There?
- Where?
- Up?
- Down?
- Talking?



### Alarms

- External notification
- Auto actions
- Event forwarding



### Data Source

- SNMP and MIBS



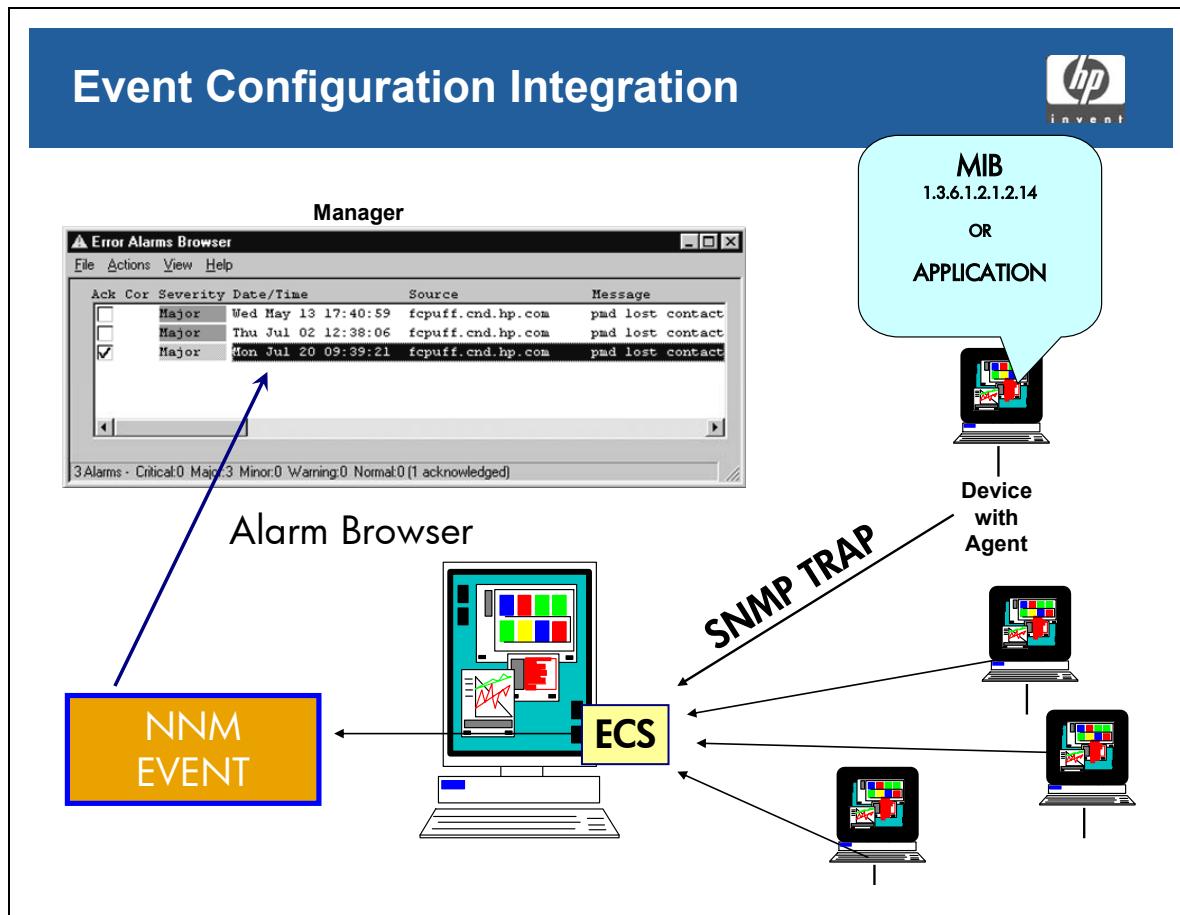
### Student Note

#### What Does NNM Provide?

HP OpenView Network Node Manager ( NNM ) provides you with a map of your network, shows you the status of objects on that network, and provides many tools for managing your network.

- Auto Discovery of network devices
- Displays "logical" network topology
- Uses standard SNMP technology
- Maintains database of network devices and status
- Allows filtering of network events
- Very flexible, many add on products, highly customizable

## B-5. SLIDE: Event Configuration Integration



### Student Note

The event comes into the NNM application management server and then within NNM you can intercept and configure that event with an NNM built in GUI tool referred to as “Event Configuration”.

- Event configuration is necessary if you want to:  
Run a program when an event happens
- Generate a trouble ticket when a node goes down or specific network alarm occurs
- Process a specific SNMP trap received from an application or take any action ... when this event or alarm occurs.

This SNMP Trap can come from the SNMP Agent on a Server, on a Network device such as a bridge, hub, router, switch, or it may come from an application to your Central OV management server. Once in OV...the possibilities are endless.

You might have one single event on the net that generates many nuisance alarms. With event correlation Services ECS on the management server, NNM can correlate many hundreds or thousands of events into a single root cause event.

## **B-6. SLIDE: OpenView Operations Integration**

### **OpenView Operations Integration**



**Monitors systems, applications, service "operations"**

#### **Management Server**

- HP-UX, Solaris, Windows ( with OVOW )

#### **Agents**

- HPUX, Solaris, NT/2K, Sinix, TRU 64 Unix, SCO, RedHat Linux, Olivetti, Novell Netware, NCR, HP MPE/iX, Irix, Pyramid, IBM AS/400, Bull

#### **Utilizes SPIs ( Smart Plug Ins )**

#### **Service Management ( Openview Navigator )**

#### **Unix and Windows Management Server:**

- Functionally are the same.
- The Agent technology is the same.
- Products can work together for enterprise level scalability.

### **Student Note**

Openview Operations (OVO) is an OV product that is used to manage, systems, services, performance, and overall “operations”.

Available on UNIX and Windows. The UNIX and Windows platform are functionally the same and utilize the same agent technology but there are some differences in the way the platform is setup and configured.

Both products can forward messages and alarms to each other to form a enterprise scaling framework. OVO can also receive SNMP traps directly or SNMP traps may be forwarded by NNM. Let us begin by taking a look at the two different platforms on which OVO is available ...

The operations staff can use a remote user interface that is not operating system specific for both platforms. There is a very robust “message browser” available in OVO with some added features and functionality. Some customers will choose to make OVO their single pane of glass and integrate all events, alarms, traps, and performance alarms into this single browser. How many of you are using OVO as a single pane of glass or common interface for your operations?

## B-7. SLIDE: OpenView Operations Agents

### OpenView Operations Agents

**Configurable**  
**Proactive**  
**Independent of manager**  
**Many server OSs:**

- HPUX, Solaris, NT/2K, Sinix, DEC Unix, SCO, RedHat, Linux, Olivetti, Novell Netware, NCR, HP MPE/iX, Irix, Pyramid, IBM AS/400, Bull

**Performance Alarms and Data Collection ( 7.X )**

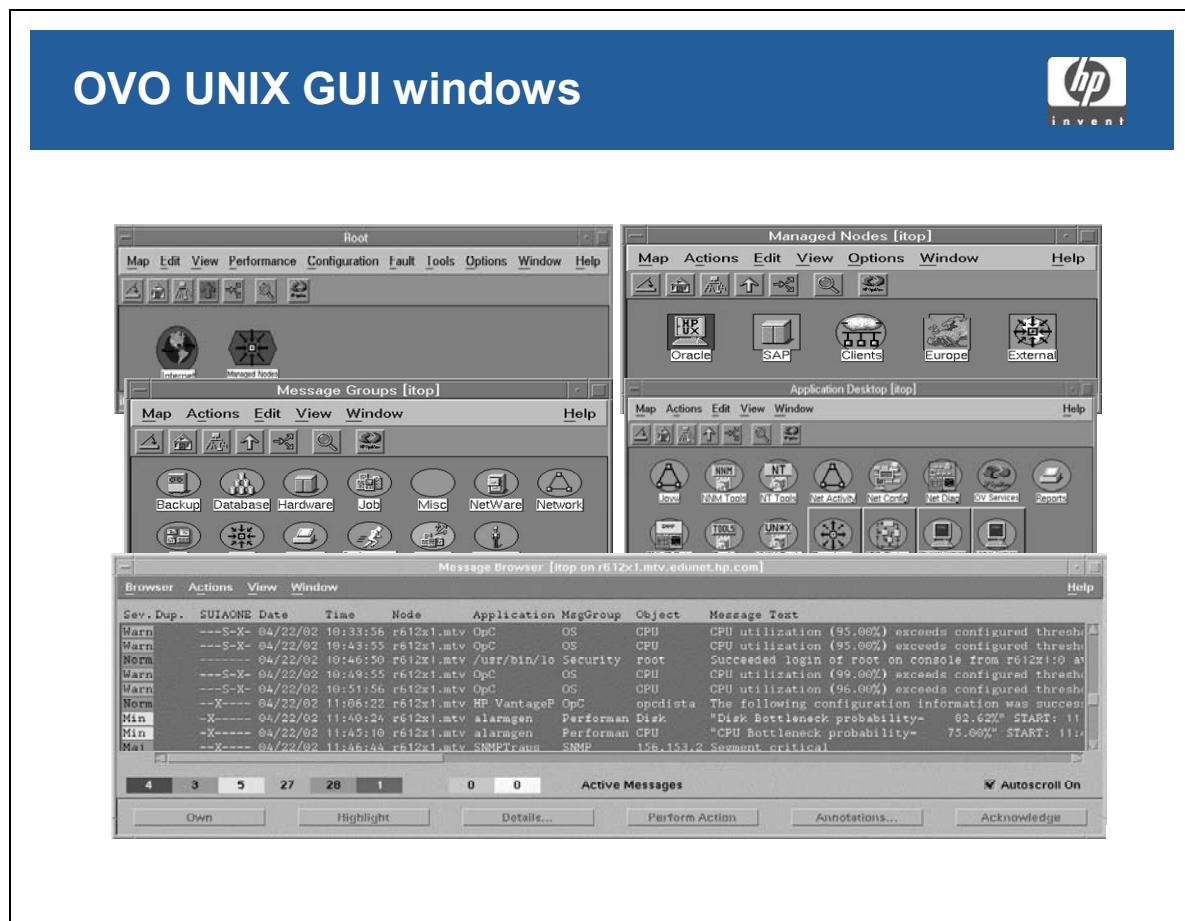
### Student Note

The source of data for OVO is provided by Agents. These OVO agents can be installed from the central management station and be pushed out to the managed node.

For example:

- Install the “OVO agent on a management server
- Create instructions for that Agent, telling what to manage
- You create instructions for an agent in UNIX via a template/policy.
- You create instructions for an agent in Windows via a policy.
- Once instructions ( templates/policies ) are created on the management server, the OVO administrator can Push them out to the agent residing on the managed node.
- If and when (and only when) there is an alarm ( based on your instructions ), the agent will send the event or alarm to the management server. The management system will not have to poll the node for this information because the agent is collecting and monitoring at the local level.
- OVO is also a recipient for SNMP traps and many other data sources as we will see.

## B-8. SLIDE: OVO Unix GUI Windows



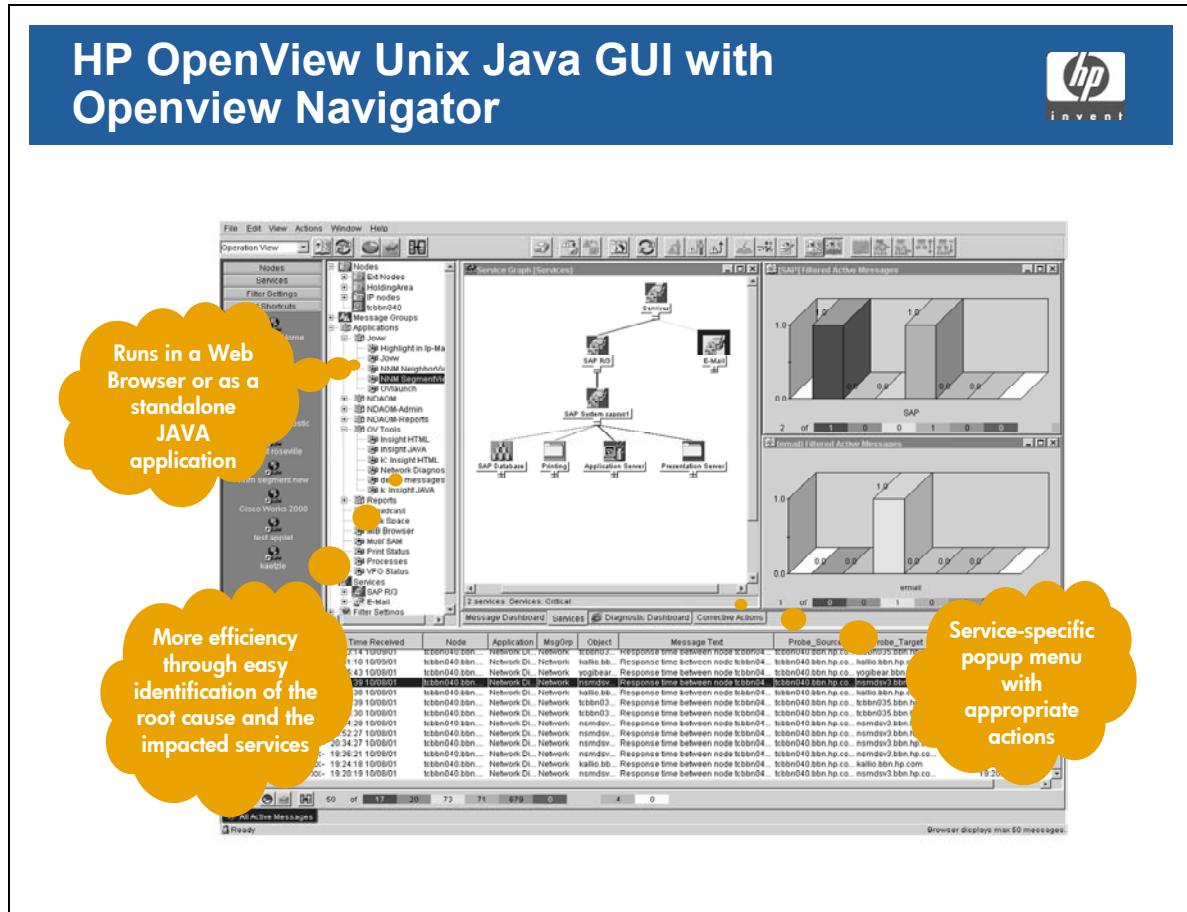
### Student Note

There is an OVO Admin interface as shown here for Openview Operations on UNIX. Alarms are generated by the Agents and displayed in the message browser. The affected nodes and message groups reflect the severity of the messages received.

A Java interface is available for operators only.

There are several indicators and each operator may have a different view of the world based on their job role or function. The customization for each operator is based on login ID, not on which interface you choose.

## B-9. SLIDE: HP OpenView Unix Java GUI With Openview Navigator



### Student Note

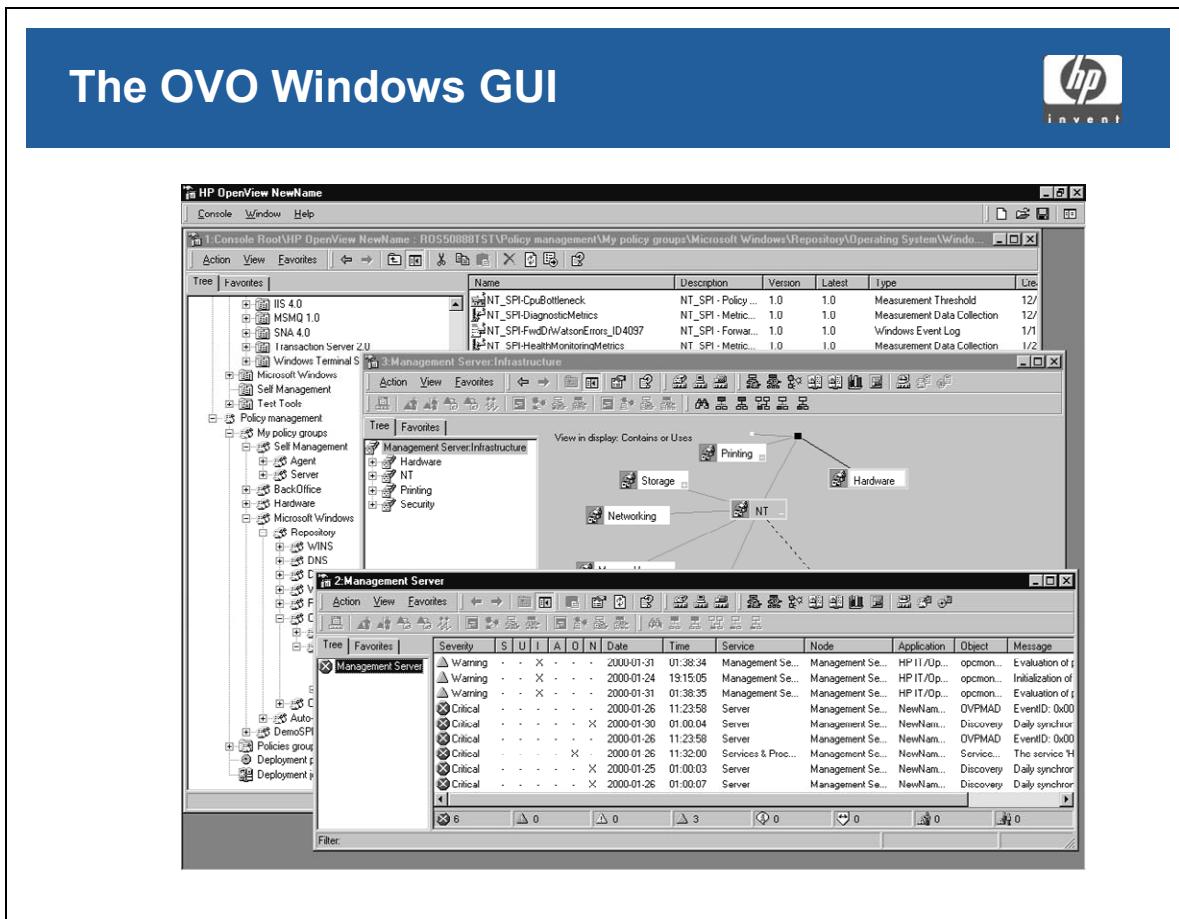
The operators have the option of using the very user friendly Java interface. The Java GUI is read only, and cannot be used for configuring or for customizing OVO.

OpenView Navigator will display only in the Java interface for OVO. Messages in the browser are related to a service. This provides a management view of the impact that an alarm has on a service.

The Service Navigator application now integrates the severity of impact for the event on one or more services in your infrastructure. This allows the operator to see the severity of an event on a service and then to drill down to the root cause of that service impact.

An SNMP trap from NNM or other source could come into OVO. OVO will propagate that to Service Navigator showing you all of the services that SNMP alarm has affected and the severity of that event on each service. This single source of data has now integrated or traversed through three different OV products.

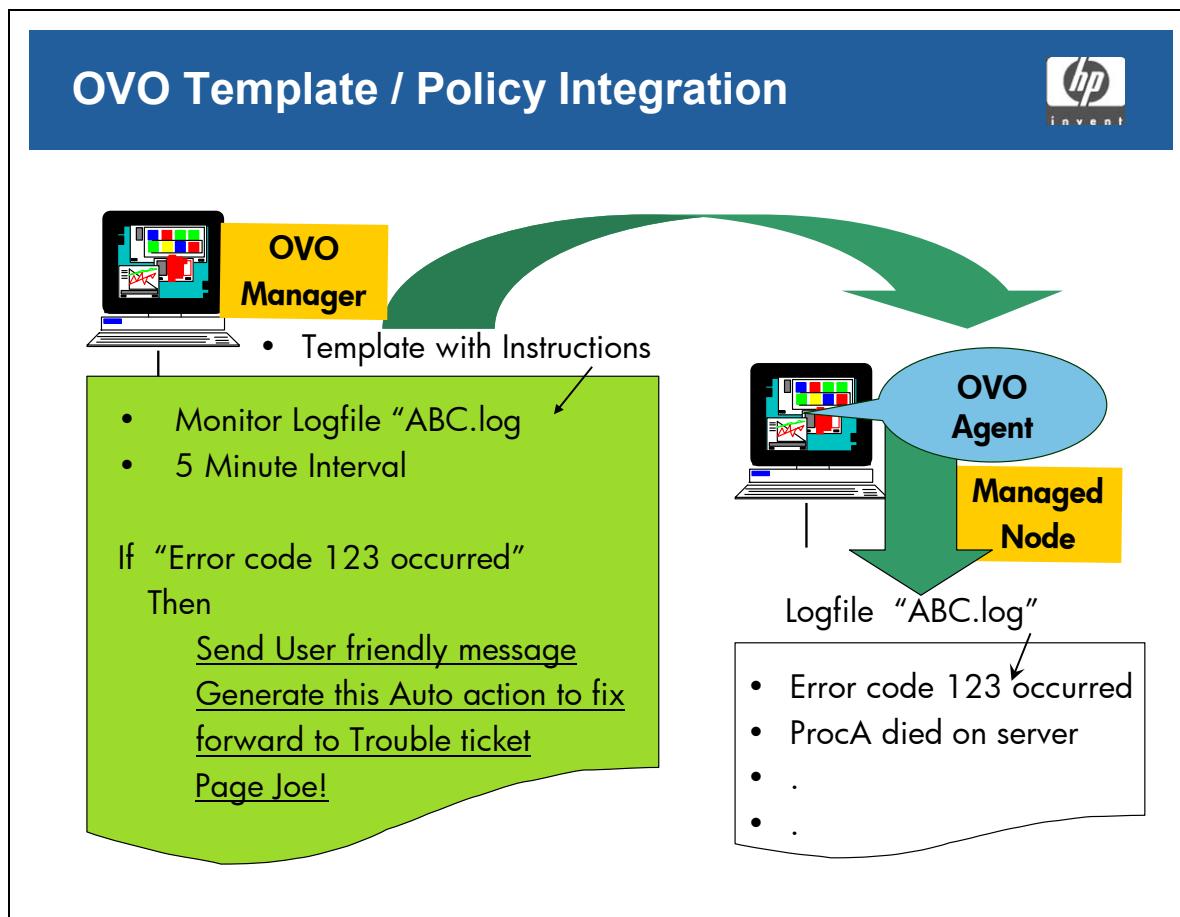
## B-10. SLIDE: The OVO Windows GUI



### Student Note

This shows the Openview Operations for Windows user interface. Functionally, Openview Operations for Unix and Windows are the same product and can interoperate very effectively but are oriented towards different communities of IT departments.

## B-11. SLIDE: OVO Template / Policy Integration



### Student Note

It is important to note that logfile encapsulation (monitoring log files) is only one of many sources of data that is available in OVO. All of the data sources are configured through the GUI for templates.

## **B-12. SLIDE: Sources of Data ... OVO**

### **Sources Of Data ... OVO**



#### **LogFile**

- Check messages from OS or application logfiles
- OVO comes with syslog.log policy that includes MC/ServiceGuard Messages

#### **Monitor**

- Run programs or MIB and check value
- OVO Includes policies to monitor processes like cmcld,

#### **Schedule**

- Run Program at scheduled interval, generate message on success, failure or both

#### **SNMP trap – collect alarms from other programs like NNM**

#### **Event Correlation – ECS circuit**

#### **OPCMSG**

- API or command line

#### **WMI Policy ( Windows only )**

### **Student Note**

This slide lists the types of data sources for OVO and the templates that can be used to manage systems and applications. Many default templates are shipped with OVO for each operating system and groups of templates included in “Smart Plug-ins” are available for Operating systems and applications.

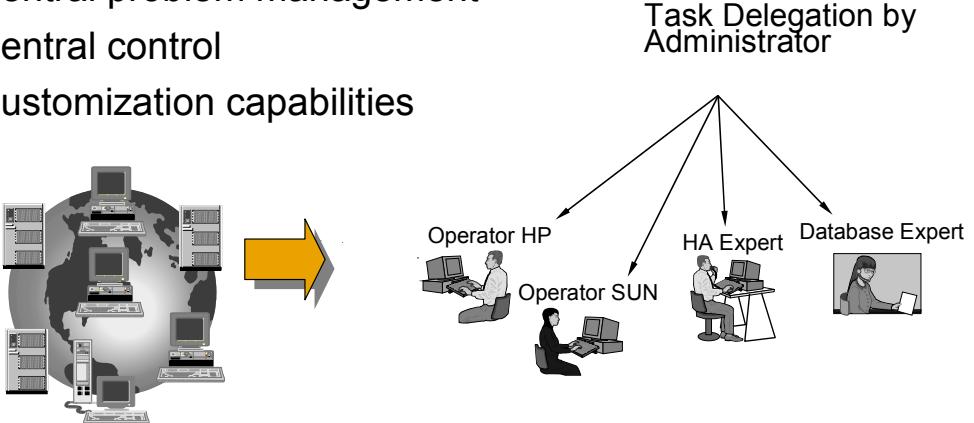
The term templates/policies is used in the UNIX platform and they are called policies on the Windows platform.

## B-13. SLIDE: OpenView Operations and Operators

### Openview Operations and Operators



- Central message management
- Central monitoring
- Workspace manager – Operators can run Serviceguard Manager application
- Central problem management
- Central control
- Customization capabilities



Task Delegation by Administrator

### Student Note

HP Openview Operations improves the productivity and management of multi-vendor distributed computing environments by providing:

- Central message management for consolidation, simplification and automation of message processing.
- Central monitoring to become more pro-active.
- Central problem management for notification, problem resolution, guidance, and tracking.
- Central control over distributed systems from a single console.
- Workspace manager to provide management information and capabilities according to the individual user's skills and responsibilities.
- Flexible customization capabilities for the administrator.

## **B-14. SLIDE: Other OpenView Products**

### **Other Openview Products**



- **Openview Performance Agent**
  - Collect OS Performance metrics every 5 minutes
  - Collect Process level metrics
  - Generate alarms with thresholds
- **Openview Performance Manager**
  - Graph OV Performance Agent metrics
- **Openview Internet Services**
  - Monitor and watch Application performance via simulation
  - Serviceguard applications can include DB and web servers
- **Data Protector**
  - Backup systems or application data.

Note: Many Openview products on HPUX can be configured as a package in a Serviceguard configuration.

### **Student Note**

The Openview Performance Agent (formerly called Measureware) is a performance data collection tool that collects over 400 metrics every 1 or 5 minutes and is available for a series of platforms including HP-UX, Solaris, Linux, Windows, and AIX. This provides a consistent application and Operating system data collection technology across platforms.

The Openview Performance Manager is a graphing and data visualization tool for the Openview Performance Agent as well as the Agent for Openview Operations. It runs on HP-UX, Solaris and Windows.

Openview Internet Services is an application performance tool that collects via simulation data on web servers, web transactions, DNS servers, ODBC Servers, and other standard protocols. If your application packages that Serviceguard works with encompass standard protocols then applications can be monitored via OVIS.

Openview Data Protector backs up systems and application data on systems. The Virtual IP used by a Serviceguard package can also be used to define backups.

**Appendix B**  
**Managing Serviceguard with HP OpenView**

Many Openview products can also be setup as a Serviceguard package. These products include: Network Node Manager, Openview Operations for Unix, Data Protector, and Openview Performance Insight. This allows these applications to be highly available.

**Appendix B**  
**Managing Serviceguard with HP OpenView**

---

## **Appendix C — Serviceguard Worksheets**

## **C-1. SLIDE: Serviceguard Worksheets**

### **Serviceguard Worksheets**



**For your use, the following worksheets are presented in this appendix:**

- Hardware Configuration worksheet
- Power Supply Configuration worksheet
- Volume Group and Physical Volume worksheet
- Cluster Configuration worksheet
- Package Failover Behavior worksheet
- Package Configuration worksheet

### **Student Notes**

For your use, we have prepared the following worksheets. They may be used as planning instruments as you prepare to configure one or more Serviceguard clusters.

## Blank Hardware Worksheet

=====

**SPU Information:**

S800 Host Name \_\_\_\_\_ S800 Series No \_\_\_\_\_

Memory Capacity \_\_\_\_\_ Number of I/O Slots \_\_\_\_\_

=====

**LAN Information:**

Name of Subnet _____	Name of Interface _____	Node IP Addr _____	Traffic Type _____
----------------------	-------------------------	--------------------	--------------------

Name of Subnet _____	Name of Interface _____	Node IP Addr _____	Traffic Type _____
----------------------	-------------------------	--------------------	--------------------

Name of Subnet _____	Name of Interface _____	Node IP Addr _____	Traffic Type _____
----------------------	-------------------------	--------------------	--------------------

=====

**Serial (RS232) Heartbeat Interface Information:**

Node Name \_\_\_\_\_ RS232 Device File \_\_\_\_\_

Node Name \_\_\_\_\_ RS232 Device File \_\_\_\_\_

=====

**X.25 Information:**

OTS subnet \_\_\_\_\_ OTS subnet \_\_\_\_\_

=====

**Disk I/O Information for Shared Disks:**

Bus Type _____	Slot Number _____	Address _____	Disk Device File _____
----------------	-------------------	---------------	------------------------

Bus Type _____	Slot Number _____	Address _____	Disk Device File _____
----------------	-------------------	---------------	------------------------

Bus Type _____	Slot Number _____	Address _____	Disk Device File _____
----------------	-------------------	---------------	------------------------

Bus Type _____	Slot Number _____	Address _____	Disk Device File _____
----------------	-------------------	---------------	------------------------

Attach a printout of the output from the ioscan -fnC disk command after installing disk hardware and rebooting the system. Mark this printout to indicate which physical volume group each disk belongs to.

**Appendix C**  
**Serviceguard Worksheets**

**Blank Power Supply Worksheet**

=====

**SPU Power:**

S800 Host Name \_\_\_\_\_ Power Supply \_\_\_\_\_

S800 Host Name \_\_\_\_\_ Power Supply \_\_\_\_\_

=====

**Disk Power:**

Disk Unit \_\_\_\_\_ Power Supply \_\_\_\_\_

=====

**Tape Backup Power:**

Tape Unit \_\_\_\_\_ Power Supply \_\_\_\_\_

Tape Unit \_\_\_\_\_ Power Supply \_\_\_\_\_

=====

**Other Power:**

Unit Name \_\_\_\_\_ Power Supply \_\_\_\_\_

Unit Name \_\_\_\_\_ Power Supply \_\_\_\_\_

## **Blank Volume Group and Physical Volume Worksheet**

=====

Volume Group Name: \_\_\_\_\_

Name of First Physical Volume Group: \_\_\_\_\_

Physical Volume Name: \_\_\_\_\_

Name of Second Physical Volume Group: \_\_\_\_\_

Physical Volume Name: \_\_\_\_\_

**Appendix C**  
**Serviceguard Worksheets**

**Cluster Configuration Worksheet**

=====

Name and Nodes:

=====

Cluster Name: \_\_\_\_\_

Node Names: \_\_\_\_\_

Maximum Configured Packages: \_\_\_\_\_

Cluster Volume Groups: \_\_\_\_\_

=====

Subnets:

=====

Heartbeat IP Addresses: \_\_\_\_\_

Non-Heartbeat IP Addresses: \_\_\_\_\_

=====

Cluster Lock Volume Groups and Volumes:

=====

First Lock VG:	First Lock Physical Volume:
_____	Name (Node 1): _____ Disk Unit No: _____ Power Unit: _____
	Name (Node 2): _____ Disk Unit No: _____ Power Unit: _____
Second Lock VG:	Second Lock Physical Volume:
_____	Name (Node 1): _____ Disk Unit No: _____ Power Unit: _____
	Name (Node 2): _____ Disk Unit No: _____ Power Unit: _____

=====

Timing Parameters:

=====

Heartbeat Interval: \_\_\_\_\_

Node Timeout: \_\_\_\_\_

Network Polling Interval: \_\_\_\_\_

Autostart Delay: \_\_\_\_\_

## Package Configuration Worksheet

=====

Package Configuration File Data:

=====

Package Name: \_\_\_\_\_

Primary Node: \_\_\_\_\_

Failover Policy: \_\_\_\_\_ Failback Policy: \_\_\_\_\_

Primary Node: \_\_\_\_\_

First Failover Node: \_\_\_\_\_

Additional Failover Nodes: \_\_\_\_\_

Package Run Script: \_\_\_\_\_ Timeout: \_\_\_\_\_

Package Halt Script: \_\_\_\_\_ Timeout: \_\_\_\_\_

Package Switching Enabled? \_\_\_\_\_ Local Switching Enabled? \_\_\_\_\_

Node Failfast Enabled? \_\_\_\_\_

Additional Package Resource:

Resource Name: \_\_\_\_\_ Polling Interval \_\_\_\_\_ Resource UP Value \_\_\_\_\_

=====

Package Control Script Data:

=====

VG[0] \_\_\_\_\_ LV[0] \_\_\_\_\_ FS[0] \_\_\_\_\_ FS\_MOUNT\_OPT[0] \_\_\_\_\_

VG[1] \_\_\_\_\_ LV[1] \_\_\_\_\_ FS[1] \_\_\_\_\_ FS\_MOUNT\_OPT[1] \_\_\_\_\_

VG[2] \_\_\_\_\_ LV[2] \_\_\_\_\_ FS[2] \_\_\_\_\_ FS\_MOUNT\_OPT[2] \_\_\_\_\_

IP[0] \_\_\_\_\_ SUBNET[0] \_\_\_\_\_

IP[1] \_\_\_\_\_ SUBNET[1] \_\_\_\_\_

X.25 Resource Name \_\_\_\_\_

Service Name: \_\_\_\_\_ Run Command: \_\_\_\_\_ Retries: \_\_\_\_\_

Service Fail Fast Enabled? \_\_\_\_\_ Service Halt Timeout \_\_\_\_\_

Service Name: \_\_\_\_\_ Run Command: \_\_\_\_\_ Retries: \_\_\_\_\_

Service Fail Fast Enabled? \_\_\_\_\_ Service Halt Timeout \_\_\_\_\_

**Appendix C  
Serviceguard Worksheets**

---

## **Appendix D — Integrating VxVM and Serviceguard**

### **Objectives**

Upon completion of this module, you will be able to do the following:

- Describe the role of VxVM disks, disk groups, and volumes in a Serviceguard cluster.
- Configure a basic VxVM rootdg, disk, disk group, and volume.
- Configure a basic VxVM-based Serviceguard package.

## **D-1. SLIDE: Serviceguard Disk Space Management Overview**

### **Serviceguard Disk Space Management Overview**

**hp invent**

- Disk space in a Serviceguard cluster may be managed via ...
  - Logical Volume Manager (LVM)
  - Veritas Volume Manager (VxVM)
  - LVM and VxVM (in the same cluster/package, but not on the same disk)
- LVM and VxVM provide similar features and functionality

The diagram illustrates a two-node Serviceguard cluster named 'cluster1'. It consists of two server nodes, 'node1' and 'node2', and a 'quorum server'.  
- **Node 1 (pkg1):** Contains a 'VxVM disk group' with three cylinders and an 'LVM volume group' with three cylinders.  
- **Node 2 (pkg2):** Contains an 'LVM volume group' with three cylinders and an 'empty VxVM rootdg disk group' (indicated by three empty cylinders).  
- **Quorum server:** Contains one cylinder labeled 'vg00'.

### **Student Notes**

Disk space in a Serviceguard cluster may be managed using Logical Volume Manager ( LVM ), Veritas Volume Manager ( VxVM ), or some combination of the two.

The graphic on the slide depicts a two node cluster. One node has a VxVM `rootdg` boot disk. The second node has an LVM `vg00` boot disk. The second node also has an empty VxVM `rootdg` disk group. A `rootdg` disk group is required on any system running VxVM.

The cluster currently has two packages: `pkg1` and `pkg2`. One package appears to store its data in a VxVM disk group, and one package appears to store its data in an LVM volume group.

Though `pkg1` is currently running on `node1` and `pkg2` is currently running on `node2`, the failure of either node would cause both packages and their associated disk/volume groups to be adopted by the other node.

## **Appendix D**

# **Integrating VxVM and Serviceguard**

Both LVM and VxVM provide online management, mirroring, striping, and other basic disk space management features. LVM has been available on HP-UX for over a decade, and continues to be the preferred disk space management solution for many cluster administrators. Some cluster administrators appreciate some of the more advanced cluster-friendly features provided by VxVM and have adopted this newer disk space management solution.

---

## D-2. SLIDE: VxVM Terminology Overview

### VxVM Terminology Overview



LVM	VxVM
Physical Volume	Disk Media
Volume Group	Disk Group
Logical Volume	Volume
Mirror	Data Plex
MWC/MCR	Log Plex
Extents	Subdisk
PVRA/VGRA	Private Region

### Student Notes

Although these disk space management solutions differ in some respects, they share many similarities. The table on the slide introduces the VxVM equivalents to traditional LVM terms and concepts.

A disk that has been initialized for use in LVM is known as an LVM **Physical Volume**. Similarly, VxVM **Disk Media** objects represent disks that have been initialized for use in VxVM.

An LVM **Volume Group** is a collection of logical volumes and physical volumes that are managed together as a unit. Similarly, a VxVM **Disk Group** is a collection of volumes, disks, and other objects that are managed together as a unit.

An LVM **Logical Volume** is a virtual disk partition that may be configured as file system, swap, or raw application space. **Volumes** serve a similar purpose in VxVM.

Each **Mirror** associated with an LVM logical volume contains a complete image of the volume's data. Similarly, each **Data Plex** associated with a VxVM volume contains a complete image of the volume's data.

LVM uses the Mirror Write Cache / Mirror Consistency Record ( **MWC/MCR** ) structures to track pending write requests to mirrored volumes. After a system crash, LVM uses the information in the MCR to determine which portions of the system's logical volumes may require resynchronization. VxVM uses a similar structure called a **Log Plex** to track pending write requests to redundant volumes. After a system crash, VxVM uses these log plexes to determine which portions of the redundant volumes may require resynchronization.

LVM subdivides physical volumes into **Physical Extents**. Similarly, VxVM subdivides disks into contiguous swaths of disk space called **Subdisks**. Note, however, that the LVM extent size is set on a per volume group basis and doesn't change from disk to disk. VxVM subdisk sizes vary according the subdisk's usage. One subdisk in a disk group may be 1K. Another subdisk on the same disk may be multiple gigabytes.

Every LVM physical volume has **PVRA/VGRA** headers that store configuration information for the disk's volume group. Similarly, every VxVM VM disk has a **Private Region** that contains configuration information for the disk's disk group.

---

### **D–3. SLIDE: VxVM Command Overview**

<b>VxVM Command Overview</b>	
<b>LVM</b>	<b>VxVM</b>
<b>pvcreate</b>	<b>vxdisksetup</b>
<b>pvremove</b>	<b>vxdiskunsetup</b>
<b>vgcreate</b>	<b>vxdg init</b>
<b>vgextend</b>	<b>vxdg adddisk</b>
<b>vgreduce</b>	<b>vxdg rmdisk</b>
<b>vgremove</b>	<b>vxdg destroy</b>
<b>vgexport</b>	<b>vxdg deport</b>
<b>vgimport</b>	<b>vxdg import</b>
<b>lvcreate</b>	<b>vxassist make</b>
<b>lvremove</b>	<b>vxassist remove volume</b>
<b>lvextend</b>	<b>vxresize</b>
<b>lvreduce</b>	<b>vxresize</b>
<b>pvdisplay/vgdisplay/lvdisplay</b>	<b>vxdisk/vxprint</b>
<b>pvmove</b>	<b>vxevac</b>



### **Student Notes**

The slide above provides a side-by-side comparison of the basic LVM/VxVM commands required to configure and manage basic VxVM disk groups and LVM volume groups. Although the command syntax varies, the concepts underlying the commands are relatively similar. The slides that follow explore these VxVM commands in greater detail.

## D-4. SLIDE: Serviceguard/VxVM Software Overview

### Serviceguard/VxVM Software Overview



- Base-VXVM, the core VxVM bundle, is included in every HP-UX 11i OE
- Serviceguard is included in the HP-UX 11i Mission Critical OE
- All of these products are also available on the HP-UX Applications DVD

Bundle	Codeword?	Comment
Base-VXVM	No	<ul style="list-style-type: none"> <li>• Included in all HP-UX Operating Environments</li> <li>• Provides basic VxVM functionality, plus rootdg mirroring</li> </ul>
B9116AA	Yes	<ul style="list-style-type: none"> <li>• VxVM “Full” product</li> <li>• Required for mirroring/RAID5 in data disk groups</li> </ul>
B9117AA	Yes	<ul style="list-style-type: none"> <li>• VxVM “Cluster Volume Manager (CVM)” bundle</li> <li>• Required for disk group shared activation</li> </ul>
B9118AA	Yes	<ul style="list-style-type: none"> <li>• VxVM “Fast Resync Option” bundle</li> <li>• Enables fast mirror resync after splitting/merging mirrors</li> </ul>
B3935DA	Yes	<ul style="list-style-type: none"> <li>• Serviceguard product</li> <li>• Required on all nodes in a cluster</li> </ul>
B8467BA	No	<ul style="list-style-type: none"> <li>• Serviceguard “Quorum Server” product</li> <li>• Only required on the quorum server</li> </ul>

### Student Notes

The table on the slide describes the software products typically found in a Serviceguard or VxVM installation.

#### VxVM Software

The VxVM suite includes several components.

- Base VxVM provides the functionality necessary to create, import, and deport disk groups, and create, extend and reduce simple and striped volumes. Base-VXVM is included in all of the HP-UX 11i operating environments.
- The “Full” VxVM product bundle ( B9116AA ) adds the ability to create mirrored and RAID 5 volumes. B9116AA requires an additional license. Customers who plan to use array-based RAID functionality rather than VxVM software-based RAID to provide redundancy don’t need this product.
- The Veritas “Cluster Volume Manager” product bundle (B9117AA) makes it possible to access a VxVM disk group read-only from one node in a cluster, while another node is concurrently making changes to the data in the disk group. B9117AA requires an

## **Appendix D**

### **Integrating VxVM and Serviceguard**

additional license. Some customers use this functionality to do online, off-host backups. CVM is beyond the scope of this discussion.

- The VxVM “Fast Resync Option” bundle (B9118AA) dramatically improves performance when doing “snapshot” online backups with VxVM. B9118AA requires an additional license. Customers who plan to use array-based online backup functionality rather than VxVM’s snapshot volume online backup functionality don’t need this product.

### **Serviceguard Software**

There are two Serviceguard bundles.

- B3935DA contains the Serviceguard product. This product must be installed on all nodes in the cluster.
- B8467BA contains the QuorumServer product. This product only needs to be installed on the quorum server.

The **swlist** command may be used to determine which of these products are installed on your system. If you installed Serviceguard as part of the Mission Critical Operating Environment, you may not have the B3935DA bundle wrapper – look for the Serviceguard *product* rather than the B3935DA *bundle*.

```
# swlist -l product ServiceGuard QuorumServer
# swlist -l bundle  Base-VXVM B9116AA B9117AA B9118AA B3935AA
```

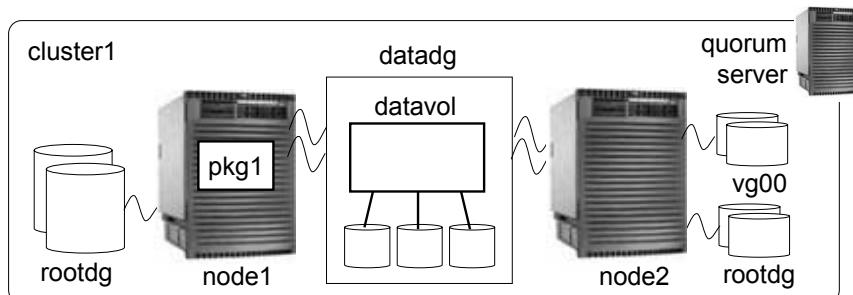
To install the products, mount the applications DVD and run **swinstall**.

## D-5. SLIDE: Cookbook Overview

### Cookbook Overview



- The remaining slides provide a cookbook for configuring the cluster shown below
- To learn more about VxVM, attend HP's H7085S course
- To learn more about Serviceguard, attend HP's H6487S course



The diagram illustrates a cluster configuration. It features two nodes, node1 and node2, each represented by a server rack icon. Node1 contains a package labeled 'pkg1'. A 'rootdg' volume is connected to both nodes. A 'datadg' volume is also connected to both nodes, containing a 'datavol' which is striped across three physical volumes. A separate 'quorum server' is shown, connected to both nodes and containing a 'vg00' volume, which in turn contains a 'rootdg' volume.

### Student Notes

The remaining slides provide a cookbook for configuring the simple cluster shown on the slide.

Note that this discussion only covers only the key features of VxVM and Serviceguard required to integrate the two products

- To learn more about VxVM mirroring, striping, recovery, performance, maintenance, and other issues, attend HP's H7085S VxVM course.
- To learn more about Serviceguard cluster and package configuration, attend HP's H6487S course.

## **D-6. SLIDE: Configure the rootdg ( no vg00 )**

### **Configure the rootdg (no vg00)**



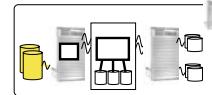
- VxVM requires one special disk group on each node called rootdg
- If you are cold-installing HP-UX, Ignite-UX can configure rootdg as the boot disk
- For improved availability, mirror the rootdg boot disk

Cold install the rootdg boot disk using VxVM/VxFS

```
/opt/ignite/bin/itool ()  
+-----+  
| Basic | Software | System | File System | Advanced |  
+-----+  
Configurations: [ HP-UX B.11.23 Default->]  
Environments: [ Mission Critical OE-64bit->]  
Root Disk: HP_36.4GMAS3367NC, 0/1/1/1.2.0, 34732 MBM  
File System:  
+-----+  
| VERITAS Volume Manager(VxVM) with VxFS  
| Whole disk with VxFS  
| Logical Volume Manager(LVM) with VxFS  
+-----+
```

Mirror the VxVM boot disk

```
node1# /etc/vx/bin/vxrootmir -v -b c0t0d0
```



### **Student Notes**

Every system that uses VxVM must have a **rootdg** disk group with at least one disk. The **rootdg** is intended to store the operating system files and directories. If you prefer to configure your boot disk as an LVM volume group, **rootdg** will likely be empty, but must exist nonetheless in order to start the VxVM daemons.

If you are cold installing your system and wish to configure the boot disk via VxVM, select “VERITAS Volume Manager” from the Ignite File System pulldown menu. This creates a VxVM boot disk with the following VxVM volumes:

```
# bdf  
Filesystem          kbytes    used   avail %used Mounted on  
/dev/vx/dsk/rootdg/rootvol  409600  205816  202216  50% /  
/dev/vx/dsk/rootdg/standvol 307200  122072  183728  40% /stand  
/dev/vx/dsk/rootdg/varvol   2609152 1098472 1500408  42% /var  
/dev/vx/dsk/rootdg/usrvol   4465976 2577312 1873976  58% /usr  
/dev/vx/dsk/rootdg/tmpvol   204800   8496  194776   4% /tmp  
/dev/vx/dsk/rootdg/optvol   3660208 2154464 1494000  59% /opt  
/dev/vx/dsk/rootdg/homevol  20480    8432  11968  41% /home
```

## **Appendix D**

### **Integrating VxVM and Serviceguard**

To ensure high availability, it's a good idea to mirror the VxVM boot disk using the automated `vxrootmir` shell script. Simply specify the `cxtxdx` name of the disk you wish to mirror to. The command works on both PARISC and Integrity servers.

```
# vxrootmir -v -b c0t0d0
vxrootmir: 10:57: Gathering information on the current VxVM root
configuration
vxrootmir: 10:57: Checking specified disk(s) for usability
vxrootmir: 10:57: Preparing disk c0t0d0 as a VxVM root disk
vxrootmir: 10:57: Disk c0t0d0 is now EFI partitioned disk c0t0d0s2
vxrootmir: 10:57: Adding disk c0t0d0s2 to rootdg as DM roottdisk02
vxrootmir: 10:57: Mirroring all volumes on root disk
vxrootmir: 10:57: Mirroring volume standvol
vxrootmir: 10:58: Mirroring volume swapvol
vxrootmir: 10:59: Mirroring volume rootvol
vxrootmir: 10:59: Mirroring volume tmpvol
vxrootmir: 10:59: Mirroring volume homevol
vxrootmir: 10:59: Mirroring volume optvol
vxrootmir: 11:01: Mirroring volume usrvol
vxrootmir: 11:03: Mirroring volume varvol
vxrootmir: 11:05: Current setboot values:
vxrootmir: 11:05: Primary:      0/1/1/1.2.0
vxrootmir: 11:05: Alternate:    <none>
vxrootmir: 11:05: Making mirror disk c0t0d0s2 (0/1/1/0.0.0) the alternate
boot disk
vxrootmir: 11:05: Disk c0t0d0s2 is now a mirrored root disk
```

## **D-7. SLIDE: Configure the rootdg (if vg00 already exists)**

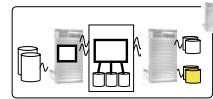
### **Configure the rootdg (if vg00 already exists)**



- If the OS is installed in vg00, VxVM requires an empty rootdg
- Configure the rootdg via the interactive vxinstall script
- For improved availability, configure two disks in rootdg

Configure an empty rootdg with two disks

```
node2# vxinstall
...
Installation options for disk c1t1d1 Menu:
VolumeManager/Install/VolumeManager/Install/c0t0d0
1      Install as a new disk. (discards data on disk!)
2      Leave this disk alone.
?      Display help about menu
??     Display help about the menuing system
q      Exit from menus
Select an operation to perform: 1
...
```



### **Student Notes**

If you already have an LVM boot disk, a `rootdg` disk group is still required if you plan to use VxVM volumes. Without a `rootdg` disk group, the `vxconfigd` daemon, which is required by VxVM, won't start. If `rootdg` is configured on a disk array, one RAID-protected LUN should be sufficient. If `rootdg` is configured on a JBOD or internal disk, however, configure two disks in `rootdg` to ensure that the `vxconfigd` daemon will still start if one of the disks in `rootdg` fails.

You can either convert your existing LVM vg00 volume group into a VxVM rootdg disk group via the `vxcp_lvmroot` command, or use two unused disks to create an empty VxVM `rootdg`. Converting a `vg00` volume group to a `rootdg` disk group is beyond the scope of this discussion.

The interactive `vxinstall` shell script provides the easiest mechanism to configure a new `rootdg`.

1. First, remove old headers from the disk( s ) you wish to configure in `rootdg`. VxVM won't overwrite existing LVM headers, even if the headers belong to a defunct volume group.

```
[node2] # dd if=/dev/zero of=/dev/rdsck/c0t0d0 bs=1024 count=1024
```

2. Run the `vxinstall` program to initialize VxVM. You only need to run this utility once, when you first install VxVM.

```
[node1] # vxinstall
```

3. When asked if you wish to enter a license key, answer "n". We will only use the "base" VxVM functionality, which doesn't require a license.

```
Are you prepared to enter a license key [y,n,q,?] n
```

4. When asked if you wish to use enclosure based names for your disks, answer "n". Enclosure-based names are most useful in disk array and SAN environments.

```
Do you want to use enclosure based names for all disks ? n
```

5. The program will interactively walk you through a series of menus. Whenever `vxinstall` asks if you wish to continue, press [Return]. If you make a mistake, you can press ^C to break out and start over again at any point.

- a) The program will search for and report your system's disk enclosures and/or controller cards. Press [Return].
- b) Next, `vxinstall` asks how you wish to configure the disks on your system. You can choose either a "Quick Installation" ( in which VxVM initializes all currently unused disks with little interaction from the administrator ) or "Custom Installation" ( in which you get to choose which disks should and shouldn't be configured ). Choose "Custom Installation".
- c) If you chose "Custom Installation", `vxinstall` should list all of the currently unused disks on the system, and ask if you wish to initialize them. You can initialize selected disks, or all of the currently unused disks. For the sake of this exercise, choose "Install all disks as new disks".
- d) Finally, VxVM gives you an opportunity to assign a meaningful name to each disk. These names will be used in place of device file names throughout the menus that follow. Accept the default disk names for the selected disks.

```
Use default disk names for these disks? [y,n,q,?] y
```

- e) When asked to confirm that you wish to destroy the selected disks, answer "y".

```
Are you sure you want to destroy the contents of this disk? y
```

- f) Proceed through the remaining confirmation screens.

## D-8. SLIDE: Configure VxVM Disks

### Configure VxVM Disks



- Disks must be initialized with VxVM headers before they can be used in a disk group

Identify available disks

```
node1# vxdisk -o alldgs list
```

Clobber any existing headers on the disks

```
node1# dd if=/dev/zero of=/dev/rdsk/c1t1d1 bs=1024 count=1024
node1# dd if=/dev/zero of=/dev/rdsk/c2t2d2 bs=1024 count=1024
node1# dd if=/dev/zero of=/dev/rdsk/c3t3d3 bs=1024 count=1024
```

Initialize the disks for use in VxVM

```
node1# /etc/vx/bin/vxdisksetup -i c1t1d1
node1# /etc/vx/bin/vxdisksetup -i c2t2d2
node1# /etc/vx/bin/vxdisksetup -i c3t3d3
```

Verify that the disks initialized successfully

```
node1# vxdisk list
```



### Student Notes

After configuring the **rootdg**, you will probably want to configure additional VxVM disks. A VxVM disk is a physical disk that has been initialized for use by VxVM. Almost any disk, from 500MB external disks to multi-terabyte HP XP disk array LUNs, can be initialized for use in VxVM.

#### Identifying Available Disks

The first step is to determine which disks are available. The **vxdisk** command is the best way to determine this. In a clustered environment, be sure to include the **-o alldgs** option. Otherwise, **vxdisk** will mistakenly indicate that disks being used by other systems on the SAN or in the cluster are unused.

Consider the sample **vxdisk -o alldgs list** output below:

```
[node1] # vxdisk -o alldgs list
DEVICE  TYPE      DISK      GROUP      STATUS
c0t0d0  simple    rootdg01  rootdg    online
c1t1d1  simple    -         -          online invalid
c2t2d2  simple    -         -          online invalid
c3t3d3  simple    -         -          simple -
-        LVM
```

You should see several columns of output:

<b>DEVICE</b>	Lists the device file names of all accessible disks.		
<b>TYPE</b>	Lists disk types. Other platforms use this column to distinguish memory-based devices from disk-based devices. VxVM on HP-UX only supports disk-based devices, so <b>TYPE</b> will always be <b>simple</b> .		
<b>DISK</b>	Reports the VM disk names for disks that have been assigned to local disk groups.		
<b>GROUP</b>	Reports which disk group each disk belongs to. If the disk group name appears in parentheses, then that indicates that the disk group isn't currently "imported" on the local system. Though these disks may be physically accessible, the disk groups that they belong to are owned by another node in your cluster, or another system on your SAN.		
<b>STATUS</b>	Reports the current status of the disk as follows:		
Online Invalid	indicates that the disk isn't being used by LVM or VxVM		
Online	indicates that the disk has been initialized for use in VxVM		
LVM	indicates that the disk has been initialized for use in LVM		

Two categories of disks may be considered candidates for inclusion in a new disk group:

- Disks that report **Online Invalid** in the **STATUS** field are usually candidates. **Online Invalid** suggests that the disk doesn't have a recognizable LVM or VxVM header. Beware, however, that disks configured using the whole disk approach will also appear to be **Online Invalid**.
- Disks that report **Online** in the **STATUS** field, and don't list a disk group name in the **GROUP** field are also candidates. These disks have VxVM headers, but haven't yet been assigned to disk groups.

### Clobbering Existing Headers

Disk that report **LVM** may or may not be available. If a disk belongs to a defunct volume group that wasn't properly removed, it will still have LVM headers, and will be reported as an LVM disk. If you are *certain* that the disk isn't currently in use, overwrite the LVM headers with the **dd** command.

```
node1# dd if=/dev/zero of=/dev/rdsck/c3t3d3 bs=1024 count=1024
```

After running **dd**, the disk should report status **online invalid**.

```
node1# vxdisk -o alldgs list
DEVICE  TYPE    DISK      GROUP      STATUS
c0t0d0  simple  rootdg01  rootdg    online
c1t1d1  simple  -         -          online invalid
c2t2d2  simple  -         -          online invalid
c3t3d3  simple  -         -          online invalid
```

### Initialize the Disks for use in VxVM

## **Appendix D**

### **Integrating VxVM and Serviceguard**

Now initialize the disks

The command required to initialize a VxVM disk is `vxdisksetup`. The `-i` (initialize) option is required.

```
[node1] # vxdisksetup -i c1t1d1
[node1] # vxdisksetup -i c2t2d2
[node1] # vxdisksetup -i c3t3d3
```

Verify your work by running `vxdisk -o alldgs list`. The disk(s) should report `online` in the STATUS column.

```
[node1] # vxdisk -o alldgs list
DEVICE  TYPE    DISK      GROUP      STATUS
c0t0d0  simple   rootdg01  rootdg    online
c1t1d1  simple   -         -          online
c2t2d2  simple   -         -          online
c3t3d3  simple   -         -          online
```

## D-9. SLIDE: Configure VxVM Disk Groups

### Configure VxVM Data Disk Groups



- A “disk group” is a group of disks that are managed together as a unit
- Disk group names are arbitrary, but typically end in “dg”
- Specify which disks you wish to include in the disk group
- Assign a “disk media name” to each disk
- Disk media names are formed by appending a unique 2-digit suffix to the dg name
- After disk group creation, disks are always referenced via their disk media names

Identify available disks  
`node1# vxdisk list`

Create the disk group  
`node1# vxdg init datadg datadg01=c1t1d1 \\\n datadg02=c2t2d2 \\\n datadg03=c3t3d3`

Verify that the disks initialized successfully  
`node1# vxdisk list`



### Student Notes

After initializing one or more VxVM disks, you can create a new disk group with the `vxdg` command.

### Disk Group Overview

A VxVM disk group is a group of disks, volumes, and other VxVM objects that are managed together as a unit. When a disk group is “deported” from a node, the disks, volumes, and other objects within that disk group become inaccessible. When a disk group is “imported” on a node, the objects in that disk group become available. When a disk group is destroyed, the objects in the disk group are destroyed as well.

One special disk group, `rootdg`, must exist on every system running VxVM.

Other disk groups may be created as is necessary. You might, for instance, create a disk group called `oracledg` for your Oracle data. You might create another disk group called `financedg` for file systems and other partitions associated with your finance department. It is recommended that you ( a ) choose intuitive names for your disk groups and ( b ) append a “dg” suffix on the end of each disk group name.

## **Appendix D**

### **Integrating VxVM and Serviceguard**

#### **Disk Media Overview**

When a disk is assigned to a VxVM disk group, it is assigned a VxVM disk media name. Typically, disk media names are formed by appending an arbitrary two-digit number on the end of the disk's disk group name. Typically the number indicates the order in which the disks were added to the disk group; not the devices hardware address or physical location on the system.

The example on the slide shows a disk group called `datadg` that has three disks:

```
datadg01  
datadg02  
datadg03
```

#### **Creating the Disk Group**

Use the `vxdg` command to create a disk group.

```
[node1] # vxdg init datadg datadg01=c1t1d1 \  
          datadg02=c2t2d2 \  
          datadg03=c3t3d3
```

The options and arguments are explained below:

<code>init</code>	<code>vxdg</code> may be used for a variety of tasks from creating disk groups, to extending, reducing, and removing disk groups. The <code>init</code> argument indicates that you wish to initialize a new disk group.
<code>datadg</code>	Specify the name of the disk group you wish to create. By convention, disk groups typically include a <b>dg</b> suffix on the end of the disk group name.
<code>datadg01=c1t1d1</code>	Specify the disk name for each disk you want to include in the disk group, as well as the <code>cxtwdx</code> name of the associated disk device. Recall that VM disk names are formed by appending a two-digit number on the end of the disk group name. At least one disk must be included in the disk group initially, but you can include others, too, if you wish.

You can verify your work with `vxdisk`.

```
[node1] # vxdisk -o alldgs list  
DEVICE   TYPE      DISK      GROUP      STATUS  
c0t0d0   simple    rootdg01  rootdg     online  
c1t1d1   simple    datadg01  datadg     online  
c2t2d2   simple    datadg02  datadg     online  
c3t3d3   simple    datadg03  datadg     online
```

## D-10. SLIDE: Configure a VxVM Volume

### Configure a VxVM Volume



- A “volume” is a virtual partition of disk space in a disk group
- A volume is typically used to store a file system, swap, or raw data
- Volume names are arbitrary, but typically end in “vol”
- VxVM supports a variety of volume layouts

Create a non-mirrored volume

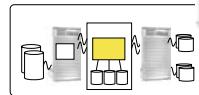
```
node1# vxassist -g datadg make datavol 16m
```

Create a mirrored volume

```
node1# vxassist -g datadg make datavol 16m layout=mirror nlog=1
```

Verify that the volume was created

```
node1# vxprint datavol
```



### Student Notes

After you initialize your disk group, you can begin creating new volumes.

#### Volume Overview

Each disk group typically contains one or more volumes. A **volume** is a VxVM object representing a virtual partition that may be used to store a file system, raw data for an application, or swap space.

A disk group may contain one or many volumes, and each volume may be anywhere from 1K, to multiple terabytes in size.

Be sure to choose descriptive volume names so other administrators can easily determine how each volume is being used. Some administrators append a `.vol` suffix on the end of each volume name, but this isn't necessary.

## **Appendix D**

### **Integrating VxVM and Serviceguard**

Volumes, like disks, are accessed via device files. Each volume has both a block and a character device file. The output below shows the device files for the `datavol` volume in the `datadg` disk group:

```
[node1] # ll /dev/vx/* /rootdg/datavol  
brw----- 1 root root 1 0x000005 Sep 4 /dev/vx/dsk/datadg/datavol  
crw----- 1 root root 54 0x000005 Sep 4 /dev/vx/rdsk/datadg/datavol
```

VxVM character device files are stored in disk group subdirectories under `/dev/vx/rdsk/`. VxVM block device files are stored in disk group subdirectories under `/dev/vx/dsk/`.

### **Creating a Simple Volume**

`vxassist` is the preferred command for creating and managing volumes. The sample command below creates a 16MB volume called `datavol` in the `datadg` disk group:

```
[node1] # vxassist -g datadg make datavol 16m
```

To ensure proper load balancing, some administrators prefer to specify which disk the volume is created on:

```
[node1] # vxassist -g datadg make datavol 16 datadg01
```

If the volume is too large to fit on one disk, you can list multiple disks on the end of the command line.

You can check your work with the `vxprint` command. Since the volume below is not mirrored, it has one data plex (`datavol-01`) which contains one subdisk (`datadg01-01`).

```
[node1] # vxprint -g datadg datavol  
TY NAME ASSOC KSTATE LENGTH Ploffs STATE Tutilo Putilo  
v datavol fsgen ENABLED 16384 - ACTIVE - -  
pl datavol-01 datavol ENABLED 16384 - ACTIVE - -  
sd datadg01-01 datavol-01 ENABLED 16384 0 - - -
```

### **Creating a Mirrored Volume**

Mirroring a volume ensures that the volume remains accessible, even if one data plex in the volume becomes inaccessible due to disk failure or other problems. In order to create a mirrored volume, you must have at least two disks in your disk group.

The sample command below creates 16MB mirrored volume called `datavol`. The `nlog=1` argument is optional, but highly recommended. It creates a log plex for the volume that ensures the plexes in the volume can be quickly resynchronized in case of a system crash.

```
[node1] # vxassist -g datadg make datavol 16m layout=mirror nlog=1
```

If you wish, you can specify which disks you want to use.

```
[node1] # vxassist -g datadg make datavol 16m layout=mirror nlog=1 \  
datadg01 datadg02 datadg03
```

## Appendix D Integrating VxVM and Serviceguard

The examples so far assume that you want to configure two plexes for the mirrored volume. VxVM actually supports up to 32 plexes per volume. Simply add the `nmirror` option to the `vxassist` command.

```
[node1] # vxassist -g datadg make datavol 16m layout=mirror \
           nmirror=2 nlog=1
```

You can check your work with the `vxprint` command. Note that mirrored volumes have multiple, equal-size plexes. This is the `vxprint` output for a volume with two data plexes and a log plex:

```
[node1] # vxprint -g datadg datavol
TY NAME      ASSOC      KSTATE    LENGTH   Ploffs  STATE Tutilo  Putilo
v  datavol    fsgen     ENABLED   16384    -       ACTIVE  -       -
pl datavol-01 datavol   ENABLED   16384    -       ACTIVE  -       -
sd datadg01-01 datavol-01 ENABLED   16384    0       -       -       -
pl datavol-02 datavol   ENABLED   16384    -       ACTIVE  -       -
sd datadg02-01 datavol-02 ENABLED   16384    0       -       -       -
pl datavol-03 datavol   ENABLED   LOGONLY  -       ACTIVE  -       -
sd datadg01-02 datavol-03 ENABLED   33       LOG     -       -       -
```

## **D-11. SLIDE: Import / Deport the Disk Group on the Other Node**

### **Import/Deport the Disk Group on the Other Node**



- Verify that the other node can import the disk group
- Unlike LVM, VxVM stores all of its disk group information in the disk headers
- Thus, there is no need to manually update /etc/lvmtab, device files, etc. on node2!

Unmount the file system and deport the disk group from the first node

```
node1# umount /data  
node1# vxvgd deport datadg
```

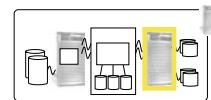
Import the disk group and mount the file system on the second node

```
node2# vxvgd import datadg  
node2# mkdir /data  
node2# mount /dev/vx/dsk/datadg/datavol /data
```

Unmount the file system and deport the disk group from the second node

```
node2# umount /data  
node2# vxvgd deport datadg
```

Leave the disk group deported in preparation for package creation



### **Student Notes**

Verify that the other node can import the disk group. Note that unlike LVM, VxVM stores all of its disk group information in the disk headers. Thus, there is no need to manually update /etc/lvmtab, device files, etc., on node2!

Unmount the file system and deport the disk group from the first node.

```
[node1] # umount /data  
[node1] # vxvgd deport datadg
```

Import the disk group and mount the file system on the second node.

```
[node2] # vxvgd import datadg  
[node2] # mkdir /data  
[node2] # mount /dev/vx/dsk/datadg/datavol /data
```

Unmount the file system and deport the disk group from the second node.

```
[node2] # umount /data
[node2] # vxdg deport datadg
```

Leave the disk group deported. It will be re-imported when the Serviceguard package is created and enabled later.

### **How Serviceguard Imports and Exports VxVM Diskgroups**

Every Serviceguard package includes a package control script which executes automatically anytime a package is started or halted on a node. If a package includes VxVM disk groups, the package's control script uses the `vxdg deport` command to deport the package's disk groups when the package is halted, and `vxdg -tfC import` to import the package's disk groups when the package is restarted.

The `-t` option specifies that the disk group should be imported with the `noautoimport` flag, which means that the disk group will not be automatically re-imported at boot time. Since disk groups included in the package control script are only imported by Serviceguard packages, they should not be auto-imported by VxVM.

The `-f` option allows the disk group to be imported even if one or more disks (a mirror, for example) isn't currently available.

The `-c` option clears any existing host ID that might be written on the disk from a prior activation by another node in the cluster. If the disk had been in use on another node which has gone down with a TOC, then the other node's host ID may still be written on the disk. This needs to be cleared so the new node's ID can be written to the disk. Note that the disk groups are not imported clearing the host ID if the host ID is set and matches a node that is not in a failed state. This is to prevent accidental importation of a disk group on multiple nodes which could result in data corruption.

---

**WARNING**

Although Serviceguard uses the `-c` option within the package control script, this option should not normally be used from the command line.

---

## **D-12. SLIDE: Manage VxVM Disk Groups**

### **Manage VxVM Disk Groups**



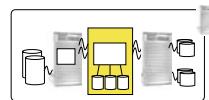
- Disk groups can be easily extended/reduced
- Unlike LVM, VxVM stores all of its disk group information in the disk headers
- Thus, there is no need to manually update /etc/lvmtab, device files, etc. on node2!

#### Add a disk to the disk group

```
node1# /etc/vx/bin/vxdisksetup -I c4t4d4  
node1# vxdg -g datadg adddisk mydg04=c4t4d4
```

#### Remove a disk from a disk group

```
node1# vxevac -g datadg mydg04 mydg03  
node1# vxdg -g datadg rmdisk mydg04  
node1# /etc/vx/bin/vxdiskunsetup c4t4d4
```



## **Student Notes**

Occasionally it may be necessary to add or remove disks in a disk group.

This task is much easier in VxVM versus LVM since all of the volume and disk group configuration information is stored in the disk headers – not in a configuration file like LVM's /etc/lvmtab file.

### **Adding a Disk to a Disk Group**

Three commands are required to add a disk to a disk group. First, identify an available disk with **vxdisk -o alldgs list** and initialize it for use by VxVM.

```
[node1] # vxdisk -o alldgs list  
[node1] # /etc/vx/bin/vxdisksetup -i c4t4d4
```

Next, add the disk to the disk group with the **vxdg** command.

```
[node1] # vxdg -g datadg adddisk datadg04=c4t4d4
```

You can verify your work with `vxdisk -o alldgs list`.

```
[node1] # vxdisk -o alldgs list
DEVICE  TYPE    DISK      GROUP      STATUS
c0t0d0  simple  rootdg01  rootdg    online
c1t1d1  simple  datadg01  datadg    online
c2t2d2  simple  datadg02  datadg    online
c3t3d3  simple  datadg03  datadg    online
```

### Removing a Disk from a Disk Group

If a disk is underutilized, it might make sense to remove the disk from its current disk group and add it to a different disk group that can use the disk more effectively. Several steps are required. First, “evacuate” all the subdisks from the disk you wish to remove. You can specify a destination disk to move the subdisks to, or you can allow `vxevac` to find a new home for the subdisks automatically. The destination disk must be in the same disk group as the disk you intend to remove. If you wish to discard the data on the disk to be removed, use the procedure described on the previous slide to remove the appropriate volumes.

```
[node1] # vxevac -g datadg datadg04 datadg03
```

Next, remove the disk from the disk group via the `vxdg` command.

```
[node1] # vxdg -g datadg rmdisk datadg04
```

Finally, if you intend to use the disk in LVM, remove the VxVM headers from the disk.

```
[node1] # /etc/vx/bin/vxdiskunsetup c4t4d4
```

You may wish to verify your work with `vxdisk -o alldgs list`. Make sure the disk is online invalid.

```
[node1] # vxdisk -o alldgs list
DEVICE  TYPE    DISK      GROUP      STATUS
c0t0d0  simple  rootdg01  rootdg    online
c1t1d1  simple  datadg01  datadg    online
c2t2d2  simple  datadg02  datadg    online
c3t3d3  simple  datadg03  datadg    online
c4t4d4  simple  datadg03  datadg    online invalid
```

## **D-13. SLIDE: Manage VxVM Volumes**

### **Manage VxVM Volumes**



- Volumes can be easily extended/reduced
- The examples below assume that you have OnlineJFS

Extend a volume

```
node1# vxassist -g datadg growto datavol 32m
```

Extend a file system

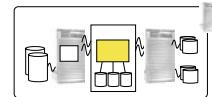
```
node1# fsadm -F vxfs -b 32M /data
```

Reduce a file system

```
node1# fsadm -F vxfs -b 16m /data
```

Reduce a volume

```
node1# vxassist -g datadg shrinkto datavol 16m
```



### **Student Notes**

#### **Extending a Volume and File System**

If a file system becomes full, it may be necessary to allocate additional space to a volume so the file system in the volume can be extended. The sample `vxassist` command below extends the `datavol` volume to 32 MB. Note that the `growto` argument specifies the final target size, *not* the amount of space that you wish to incrementally add to the volume.

```
[node1] # vxassist -g datadg growto datavol 32M
```

If you want to specify which disk the additional space should be taken from, specify a disk name on the end of the command line.

```
[node1] # vxassist -g datadg growto datavol 32M datadg01
```

In either of the commands above, you can use the **growby** argument in place of **growto** to specify the amount of space you want to add to the volume. The sample command below adds 16MB to **datavol**.

```
[node1] # vxassist -g datadg growby datavol 16M OR
[node1] # vxassist -g datadg growby datavol 16M datadg01
```

After extending the volume, extend the file system, too.

```
[node1] # fsadm -F vxfs -b 32m /data
```

**vxprint** to verify that the volume was properly extended. Note that the volume and plex sizes increased to 32 MB.

```
[node1] # vxprint -g datadg datavol
TY NAME      ASSOC      KSTATE    LENGTH Ploffs STATE   Tutilo  Putilo
v  datavol    fsgen     ENABLED  32768   -       ACTIVE  -      -
pl datavol-01 datavol   ENABLED  32768   -       ACTIVE  -      -
sd datadg01-01 datavol-01 ENABLED 32768   0       -       -      -
```

## Reducing a File System and Volume

The procedure for reducing a volume is similar to the process for extending a volume.

First, reduce the file system within the volume.

```
[node1] # fsadm -F vxfs -b 16m /data
```

Once the file system has been successfully reduced, it is safe to reduce the volume itself. The sample **vxassist** command below reduces the **datavol** volume to 16MB.

```
[node1] # vxassist -g datadg shrinkto datavol 16M
```

Alternatively, you can use the **shrinkby** argument to specify the amount of space you wish to remove from the volume. In either case, use **vxprint** to verify that the reduction succeeded.

```
[node1] # vxprint -g datadg
TY NAME      ASSOC      KSTATE    LENGTH Ploffs STATE   Tutilo  Putilo
v  datavol    fsgen     ENABLED  16384   -       ACTIVE  -      -
pl datavol-01 datavol   ENABLED  16384   -       ACTIVE  -      -
sd datadg01-01 datavol-01 ENABLED 16384   0       -       -      -
pl datavol-02 datavol   ENABLED  16384   -       ACTIVE  -      -
sd datadg02-01 datavol-02 ENABLED 16384   0       -       -      -
```

## D-14. SLIDE: Configure a Package

### Configure a Package



Create and customize a package startup script, and propagate it to all nodes.  
Be sure to add your package's disk group names and filesystem mount parameters.

Create a template package control script

```
[node1] # cmmakepkg -s /etc/cmcluster/pkg1/pkg1.sh
```

Customize the package control script

```
[node1] # vi /etc/cmcluster/pkg1/pkg1.sh
# VXVM DISK GROUPS
VXVM_DG[0]="datadg"
...
# FILESYSTEMS
LV[0]=/dev/vx/dsk/datadg/datavol
FS[0]=/data;
FS_MOUNT_OPT[0]="-o rw"
FS_TYPE[0]="vxfs"
```

Copy the package control script to the other node

```
[node1] # ssh node2 "mkdir /etc/cmcluster/pkg1"
[node1] # scp /etc/cmcluster/pkg1/pkg1.sh \
           node2:/etc/cmcluster/pkg1/
```



### Student Notes

Create a package control script for the package.

```
[node1] # cmmakepkg -s /etc/cmcluster/pkg1/pkg1.sh
Package control script is created.
This file must be edited before it can be used.
```

Edit the package control script. Look for the **VXVM\_DG[ ]** array definition lines. Create an array entry for each VxVM disk group you wish to associate with the package. When Serviceguard runs a package on a node, it will use these array entries to determine which VxVM disk groups need to be imported on the node to support the package. Also add **LV[ ]**, **FS[ ]**, **FS\_MOUNT\_OPT[ ]**, and **FS\_TYPE[ ]** array entries for each VxVM volume/file system that Serviceguard should mount to support the package. Other portions of the script may be customized to specify which network addresses and daemons/processes are included in the package. This discussion focuses only on the components required to support VxVM disk groups in Serviceguard packages.

```
[node1] # vi /etc/cmcluster/pkg1/pkg1.sh
# VXVM DISK GROUPS
VXVM_DG[0]="datadg"
...
# FILESYSTEMS
LV[0]=/dev/vx/dsk/datadg/datavol
FS[0]=/data
FS_MOUNT_OPT[0]="-o rw"
FS_TYPE[0]="vxfs"
```

Copy the control script to the other node.

```
[node1] # ssh node2 "mkdir /etc/cmcluster/pkg1"
[node1] # scp /etc/cmcluster/pkg1/pkg1.sh node2:/etc/cmcluster/pkg1/
pkg1.sh      100%   51KB   0.0KB/s   00:00
```

---

## **D-15. SLIDE: For Further Study**

### **For Further Study**



This appendix only discussed the most basic features of VxVM and Serviceguard.

Administrators who manage VxVM/Serviceguard clusters should attend:

- H7085S (Veritas Volume Manager)
- H6487S (Hands-on with Serviceguard)
- U8601S (Disaster Tolerant Architectures with Serviceguard,  
Oracle 9i, and Oracle 10g RAC)

### **Student Notes**

This discussion only covered the most basic aspects of VxVM and Serviceguard configuration. In order to successfully manage a VxVM/Serviceguard cluster, HP encourages customers to attend all three courses listed on the slide above.

## Alternate Lab: New Package Configuration : **xclock** Package

### Directions

Configure your cluster to contain a package, which displays an **xclock** on a screen.

### Part 1: Create and Edit the Package Configuration File

#### On Node1

1. Make a directory for the **xclock** package files.

#### **Answer:**

```
# cd /etc/cmcluster
# mkdir xclock
```

2. Change directory to the **xclock** package directory that was just created. Once there, create the package configuration file template.

#### **Answer:**

```
# cd xclock
# cmmakepkg -v -p xclock.conf
```

3. Next, edit the package configuration file template.

```
# vi xclock.conf
```

and modify the following parameters:

PACKAGE_NAME	xclock
NODE_NAME	<node1>
NODE_NAME	<node2>
RUN_SCRIPT	/etc/cmcluster/xclock/xclock.cntl
RUN_SCRIPT_TIMEOUT	NO_TIMEOUT
HALT_SCRIPT	/etc/cmcluster/xclock/xclock.cntl
HALT_SCRIPT_TIMEOUT	NO_TIMEOUT
SERVICE_NAME	xclock_service
SUBNET	X.X.X.X ← Use netstat -in to obtain the 10-net subnet

4. Save the changes and exit the file.

## **Part 2: Create and Edit the Package Control Script, and Distribute this Script to All Nodes**

### **On Node1**

1. Create the package control script template.

**Answer:**

```
# cd /etc/cmcluster/xclock
# cmmakepkg -v -s xclock.cntl
```

2. Edit the package control script template. Remember that this file is a script and that all variable assignments cannot contain any spaces. See the “**Special Note**” box below before starting this step.

```
# vi xclock.cntl
```

and modify the following parameters:

```
SERVICE_NAME[ 0 ]="xclock_service"
SERVICE_CMD[ 0 ]="/usr/bin/X11/xclock -update 1 -bg blue \
               -title \"Your_cluster_name -display _____\""
SERVICE_RESTART[ 0 ]="-r 2"
```

#### **Special Note**

Ask your instructor for the value that you should use as an argument to the `-display` option.

3. Save changes and exit the file.
4. Ask your instructor if you should run `xhost +` on the machine where the `xclock` will display to make sure that host will accept a window from another machine. If your instructor says to run `xhost +`, please do so.

**Answer:**

```
# xhost +           ← Only if needed.
```

5. Next, on `Node2`, create a subdirectory for your `xclock` package.

**Answer:**

```
# mkdir /etc/cmcluster/xclock
```

6. Now, back on `Node1`, make the `xclock` control script executable, and copy it to `Node2` within your cluster.

**Answer:**

```
# chmod 755 xclock.cntl
# rcp xclock.cntl <Node2>:/etc/cmcluster/xclock/.
```

7. Finally, on `Node2`, edit the `xclock.cntl` control script and change the background color of the `xclock` to **red**. Now, when a failover occurs, the color of the `xclock` will change. This change in color will indicate that the package has failed over since the control script on the second node starts the package (and starts `xclock` with a red background).

```
# cd /etc/cmcluster/xclock
# vi xclock.cntl
```

## **Part 3: Create and Distribute the Binary File, and Start the Package**

### **On Node1**

1. Check the configuration file for errors and fix any that you find.

**Answer:**

```
# cd /etc/cmcluster/xclock
# cmcheckconf -v -P xclock.conf
```

2. Apply the new configuration, which will compile and distribute a new cluster binary file.

**Answer:**

```
# cd /etc/cmcluster/xclock
# cmapplyconf -v -P xclock.conf
```

3. If the cluster is not started, start it now. If the cluster was already running, start your xclock package.

**Answer:**

```
# cmrunc1 -v
# cmmodpkg -e xclock
```

4. View the package status. We should see that the xclock package is now running on its primary node.

**Answer:**

```
# cmviewcl -v
```

5. We can test the `xclock` package behavior by doing the following:

```
# cmhaltpkg -v xclock
# cmviewcl -v

# cmrunpkg -v -n <adoptive_node> xclock
# cmmodpkg -v -e xclock

# cmviewcl -v
```

6. We can further test our package by killing the `xclock` PID. We will kill the `xclock` process PID 3 times; the first two times because we specified 2 retries, and the third time to force a failover.

Because of step 5 above, the package should now be running on `<adoptive_node>`. Execute the following steps on `<adoptive_node>` to force a failover back to `<primary_node>`.

```
# ps -ef | grep xclock           ← Kill the xclock process for the first time.
# kill <pid>                   ← The process should restart on the same node.

# ps -ef | grep xclock           ← Kill the xclock process for the second time.
# kill <pid>                   ← The process should restart on the same node.

# ps -ef | grep xclock           ← Kill the xclock process for the third time.
# kill <pid>                   ← Process should fail over to <primary node>.

# cmviewcl -v                  ← The xclock package restarts on <primary_node>.

# cmmodpkg -v -e -n <adoptive_node> xclock
# cmviewcl -v                  ← xclock is now running on <primary node>. */
```

7. The package should now be running on `<primary_node>`. Execute the following steps on `<primary_node>` to force a failover back to `<adoptive_node>`.

```
# ps -ef | grep xclock           ← Kill the xclock process for the first time.
# kill <pid>                   ← The process should restart on the same node.

# ps -ef | grep xclock           ← Kill the xclock process for the second time.
# kill <pid>                   ← The process should restart on the same node.

# ps -ef | grep xclock           ← Kill the xclock process for the third time.
# kill <pid>                   ← Process should fail over to <adoptive node>.

# cmviewcl -v                  ← The xclock package restarts on <adoptive_node>.

# cmmodpkg -v -e -n <primary_node> xclock
# cmviewcl -v                  ← xclock now running on <adoptive_node>.
```

## **Part 4: Modifying FAILBACK\_POLICY**

### **Directions**

1. Modify your `xclock` package to fail back when the primary node comes back to the cluster, and then test it.

#### **Answer:**

In the "`xclock.conf`" file, change the line that now reads ...

FAILBACK_POLICY	MANUAL
-----------------	--------

to read ...

FAILBACK_POLICY	AUTOMATIC
-----------------	-----------

and then reconfigure the package in the normal way.

```
# cmhaltcl -f
# cd /etc/cmcluster/xclock
# cmapplyconf -P xclock.conf
# cmrunc1
# cmviewcl -vp xclock
# cmhaltnode -f (on primary)
# cmrunnode      (on primary)
```

2. Did the `xclock` package come back to the primary node?

#### **Answer:**

Yes.

3. While there might be compelling reasons to use the AUTOMATIC fallback policy, we will be performing other operations with the `xclock` package that require the MANUAL fallback policy. Therefore, modify your `xclock` package configuration file again to reflect a MANUAL fallback.

#### **Answer:**

```
# cmhaltpkg xclock
# vi /etc/cmcluster/xclock.conf
```

Change FAILBACK from AUTOMATIC to MANUAL

```
# cmcheckconf -P /etc/cmcluster/xclock.conf
# cmapplyconf -P /etc/cmcluster/xclock.conf
```

## Part 5: Adding a Package Monitoring Script to the xclock Package

### Directions

- Instead of using `/usr/bin/X11/xclock` as the service, create a separate service monitoring script called `/etc/cmcluster/xclock.xclock.mon` as follows:

```
# cd /etc/cmcluster/xclock
# vi xclock.mon
#!/usr/bin/sh

LOG="/etc/cmcluster/xclock/xclock.cntl.log"

echo "Now entering the xclock package monitor \c" >> $LOG
echo "script on $(hostname) at $(date)." >> $LOG
while true
do
    if ps -ef | grep -v grep | grep -q "/usr/bin/X11/xclock"
        then
            echo "Package xclock apparently ok \c"
            echo "on $(hostname) at $(date)."
            sleep 10
        else
            echo "Package xclock failed at $(date) \c"
            echo "from node $(hostname)."
            exit
        fi
    done >> $LOG
```

The extra echo statements in the above script are sometimes referred to as "debug lines". Once the package and its scripts are working properly, these debug lines may be safely commented out. Good programming practice would suggest that these "debug lines" would be commented out (but not removed) from the script since they may be needed again in the future.

Ensure that the monitor is executable and copy it to the other system in your cluster.

#### Answer:

```
# chmod 755 /etc/cmcluster/xclock/xclock.mon
# rcp xclock.mon <node2>:/etc/cmcluster/xclock/.
```

## Appendix E

### Alternate Package Lab

2. In the `xclock` package control script `/etc/cmcluster/xclock/xclockcntl`, change the following two lines.

```
SERVICE_CMD[0]="/etc/cmcluster/xclock/xclock.mon"
SERVICE_RESTART[0]="-r 0"
```

Serviceguard will now be watching the `xclock.mon` process, and will use zero restarts. In turn, the `xclock.mon` process will be watching the health of `xclock` itself.

3. In the `customer_defined_run_cmds` function of the `xclock` package control script, `/etc/cmcluster/xclock/xclockcntl`, remove the line beginning with the colon (`:`). Then, in its place, enter the command to start our `xclock`. Do not remove the `"test_return 51"` line.

```
/usr/bin/X11/xclock -update 1 -bg blue \
    -title Your_cluster_name -display _____ &
```

**Important:** Don't forget to put your `xclock` process in the background!

4. In the `customer_defined_halt_cmds` function, remove the line beginning with the colon (`:`). Do not remove the `"test_return 52"` line. Insert the following lines just before the `"test_return 52"` line. This will stop your `xclock` if it is running:

```
function customer_defined_halt_cmds
{
LOG="/etc/cmcluster/xclock/xclockcntl.log"

echo "Now entering the customer_defined_halt_cmds \c"          >> $LOG
echo "on $(hostname) at $(date)."                                >> $LOG
if ps -ef | grep -v grep | grep -q "/usr/bin/X11/xclock"
    then
        echo "Found the xclock process at $(date)."           >> $LOG
        echo "Killing xclock process at $(date)."           >> $LOG
        kill -9 $(ps -ef | grep -v grep | grep "/usr/bin/X11/xclock" \
            | cut -c10-14)
    else
        echo "Note: In customer_defined_halt_cmds \c"        >> $LOG
        echo "on $(hostname), and could not find the \c"      >> $LOG
        echo "xclock process at $(date)."                      >> $LOG
    fi
```

---

**NOTE:** If Serviceguard believes the halt script did not properly stop the application, the halt script will fail, and Serviceguard will not start the package on the adoptive node to avoid risking application data corruption.

---

5. Now, copy the revised package control script to node2.

**Answer:**

```
# rcp /etc/cmcluster/xclock/xclock.cntl <node2>:/etc/cmcluster/xclock
```

6. On node\_2, again edit the control script to change the xclock background color to **red**.

**Answer:**

```
# vi xclock.cntl
```

7. Now verify that all the pieces work. Use your "xclock pkg" desktop ( or open a terminal window and issue the tail -f /etc/cmcluster/xclock/xclock.cntl.log command ) to view the cluster activities.

**Answer:**

```
# tail -f /etc/cmcluster/xclock/xclock.cntl.log
```

8. Halt the cluster and then verify that the xclock program has also exited.

**Answer:**

```
# cmhaltcl -v -f
# ps -ef | grep xclock      ( if necessary, kill the PID for the xclock process )
```

9. Now, restart the cluster, which automatically restarts the xclock package.

**Answer:**

```
# cmrunc1 -v
```

**Appendix E**  
**Alternate Package Lab**

10. Kill the `xclock` process (one time) and verify that it restarts on the adoptive node.

**Answer:**

```
# ps -ef | grep xclock  
# kill <pid>  
# cmviewcl -v -p xclock
```

11. Verify that the echo message was written to the package log file on the failing node.

**Answer:**

```
# vi xclock.cntl.log
```

12. Return the `xclock` package to the primary node by halting, then immediately restarting the cluster.

**Answer:**

```
# cmhaltcl -v -f  
# cmrunc1 -v
```

## Part 6: **xclock** Package run / halt Script Modification

### Directions

1. First, recognize that the "**cluster symmetry**" that was discussed in Module 3 has been seriously violated in our **xclock** package. Remember that we are currently running two different **xclock.cnt1** files. The package control script on the primary node directs the **xclock** application to start with a blue background while the same script on the adoptive node directs the **xclock** application to start with a red background. This configuration violates cluster symmetry.

In cases like this, where different code is required for each server in the cluster, the strong recommendation is to avoid maintaining multiple versions of the run / halt script (one for each server in the cluster). Instead, use logic in the script to maintain a single version of the run/halt script that would be appropriate on all nodes in the cluster, thus maintaining a symmetric cluster. To fix this, modify the control script as follows:

```
if [ $(hostname) = "primary_node_name" ]
then
    COLOR="blue"
else
    COLOR="red"
fi
/usr/bin/X11/xclock -bg "$COLOR" -title "Your_cluster_name" \
                    -update 1 -display _____ &
```

2. Copy the control script to the adoptive node.

### Answer:

```
# rcp xclock.cnt1 <node2>:/etc/cmcluster/xclock/.
```

3. To activate the above modifications, halt, and then immediately restart our **xclock** package.

The **xclock** package should be running on the primary node.

### Answer:

```
# cmhaltpkg -v xclock
# cmrungpkg -v -n "primary_node" xclock
# cmmodpkg -v -e xclock
```

Appendix E  
**Alternate Package Lab**

4. Test the revised `xclock` package by killing the `xclock` process. The `xclock` package will start on the adoptive node.

After the package fails over to the adoptive node, re-enable the primary node to run the package.

**Answer:**

```
# ps -ef | grep xclock
# kill <pid>
# cmmodpkg -v -e -n <primary_node> xclock
```

5. To further test our revised `xclock` package, halt and restart it.

This should return the `xclock` package to its primary node.

**Answer:**

```
# cmhaltpkg -v xclock
# cmrunpkg -v xclock
# cmmodpkg -v -e xclock
```

6. Congratulations on returning your cluster to a symmetric state!