

HP Technology Forum & Expo

Get connected. People. Technology. Solutions.



Session 1818:
Achieving the
Highest Possible
Availability in Your
OpenVMS Cluster

Keith Parris
Systems/Software Engineer, HP



High-Availability Principles



Availability Metrics

- Concept of “Nines”

Availability	Avail (%)	# Nines	Downtime	Units
0.99	99%	2	3.6525	days
0.9999	99.9%	3	8.766	hours
0.99999	99.99%	4	52.596	minutes
0.999999	99.999 %	5	315.576	seconds

Know the Most Failure-Prone Components

- Things with moving parts
 - Fans
 - Spinning Disks
- Things which generate a lot of heat
 - Power supplies
- Things which are very new (not yet “burned in”)
 - “Infant Mortality” of electronics
- Things which are very old
 - Worn out

Consider Facilities and Infrastructure

- Datacenter needs:
 - UPS
 - Generator
 - Redundant power feeds, A/C
- Uptime Institute says a datacenter with best practices in place can do “4 nines” at best
 - <http://www.upsite.com/>
 - Conclusion: You need multiple datacenters to go beyond “4 nines” of availability

Eliminate Single Points of Failure

- Can't depend on anything of which there is only one
- Incorporate redundancy
- Examples:
 - DNS Server at one site in OpenVMS DT Cluster
 - Single storage box under water sprinkler
- Insufficient capacity can also represent a SPOF
 - e.g. A/C
- Inadequate performance can also adversely affect availability

Eliminate Single Points of Failure

- Even an entire disaster-tolerant OpenVMS cluster could be considered to be a potential SPOF
 - Potential SPsOF for a cluster:
 - Single SYSUAF, RIGHTSLIST, etc.
 - Single queue manager queue file
 - Shared system disk
 - Reliable Transaction Router (RTR) can route transactions to two different back-end DT clusters
 - Some customers have provisions in place to move customers (and their data) between two different clusters

Detect and Repair Failures Rapidly

- MTBF and MTTR both affect availability
 - Redundancy needs to be maintained by repairs to prevent outages from being caused by subsequent failures
 - Mean Time Between Failures quantifies how likely something is to fail
 - Mean Time To Repair affects chance of a second failure causing downtime
- Detect failures so repairs can be initiated
 - Track shadowset membership
 - Use `LAVC$FAILURE_ANALYSIS` to track LAN cluster interconnect failures and repairs

Take Advantage of Increased Redundancy Levels with OpenVMS

- OpenVMS often supports 3X, 4X, or even higher levels of redundancy
 - LAN adapters
 - Use Corporate network as backup SCSI interconnect
 - Fibre Channel HBAs and fabrics
 - 3-member shadowsets
 - Even 3-site DT clusters

Solve Problems Once and For All

- Record data about failures so they aren't repeated
 - Take crash dumps and log calls
 - Put Console Management system in place to log console output

Record Data and Have Tools in Place in Case You Need Them

- Availability Manager
 - Can troubleshoot problems even when system is hung
- Performance Data Collector
 - e.g. T4

Consider Component Maturity

- Reliability of products tends to get better over time as improvements are made
 - ECOs, patches, quality improvements
- Common advice is “Never use V1.0 of anything”
- Don’t be the first to use a new product or technology
 - Not enough accumulated experience in the field yet to find and correct the problems
- If it’s working well, don’t throw out a perfectly good “mature” product just because it’s not sexy anymore
- Don’t be the last to continue using an old product or technology past the end of its market life
 - Support goes away

Implement a Test Environment

- Duplicate production environment as closely as possible
 - Run your unique applications
 - Same exact HW/SW/FW if possible
 - Same scale if possible
- Test new things first in the Test Environment, not in the Production Environment
- Can leverage this equipment for Development / QA / Disaster Recovery to help justify cost

Software Versions and Firmware Revisions

- Stay on supported versions
 - Troubleshooting assistance is available
 - Problems you encounter will get fixed
- Don't be the first to run a new version
- Don't be the last to upgrade to a new version
- Test new versions in a test environment first

Patches and ECO Kits

- Monitor patch/ECO kit releases
 - Know what potential problems have been found and fixed
 - Consider the likelihood you will encounter a given problem
- If you require high availability, consider letting a patch “age” a while after release before installing in production
 - Avoids installing a patch only to find out it’s been recalled due to a problem
- Install patches/ECO kits in test environment to detect problems with it in your specific environment
- Avoid waiting too long to install patches/ECO kits or you may suffer a problem which has already been fixed

Managing Change

- Changes in the configuration or environment introduce risks
- There must be a trade-off and balance between keeping up-to-date and minimizing unnecessary change
- Try to change only one thing at a time
 - Makes it easier to determine cause of problems and makes it easier to back out changes

Computer Application of Biodiversity

- Using a mix of different implementations improves survivability
- Examples:
 - British Telecom's X.25 network
 - GIGAswitch/FDDI and Cisco LANs
 - Cross-over cable as cluster interconnect
 - DE500 and DE602 LAN adapters
 - Shadowing between different disk controller families

Minimize Complexity, and Number of Things Any of Which can Fail and Cause an Outage

- Examples:
 - Simplest hardware necessary
 - Minimize node count in cluster

Node Interactions in an OpenVMS Cluster



Examples Where a Problem on 1 Node in a Cluster Can Affect Another Node

- Failed node held a conflicting lock at the time it failed
- Failed node is the Lock Master node for a given resource
- Failed node is the Lock Directory Lookup node for a given resource
- Failure of a node which has a shadowset mounted could cause a shadow Merge operation to be triggered on the shadowset, adversely affecting I/O performance

Examples Where a Problem on 1 Node in a Cluster Can Affect Another Node

- Failure of a node which is Master node for a Write Bitmap may cause a delay to a write operation
- Failure of enough nodes could cause loss of quorum and cause all work to pause until quorum is restored

Examples Where a Problem on 1 Node in a Cluster Can Affect Another Node

- A node may be slow and hold a lock longer than minimum time necessary, delaying other nodes' work
- Application or user on one node might delete a file on a shared disk which is a file needed by another node
- Heavy work from one node (or environment) to a shared resource such as disks/controllers could adversely affect I/O performance on other nodes

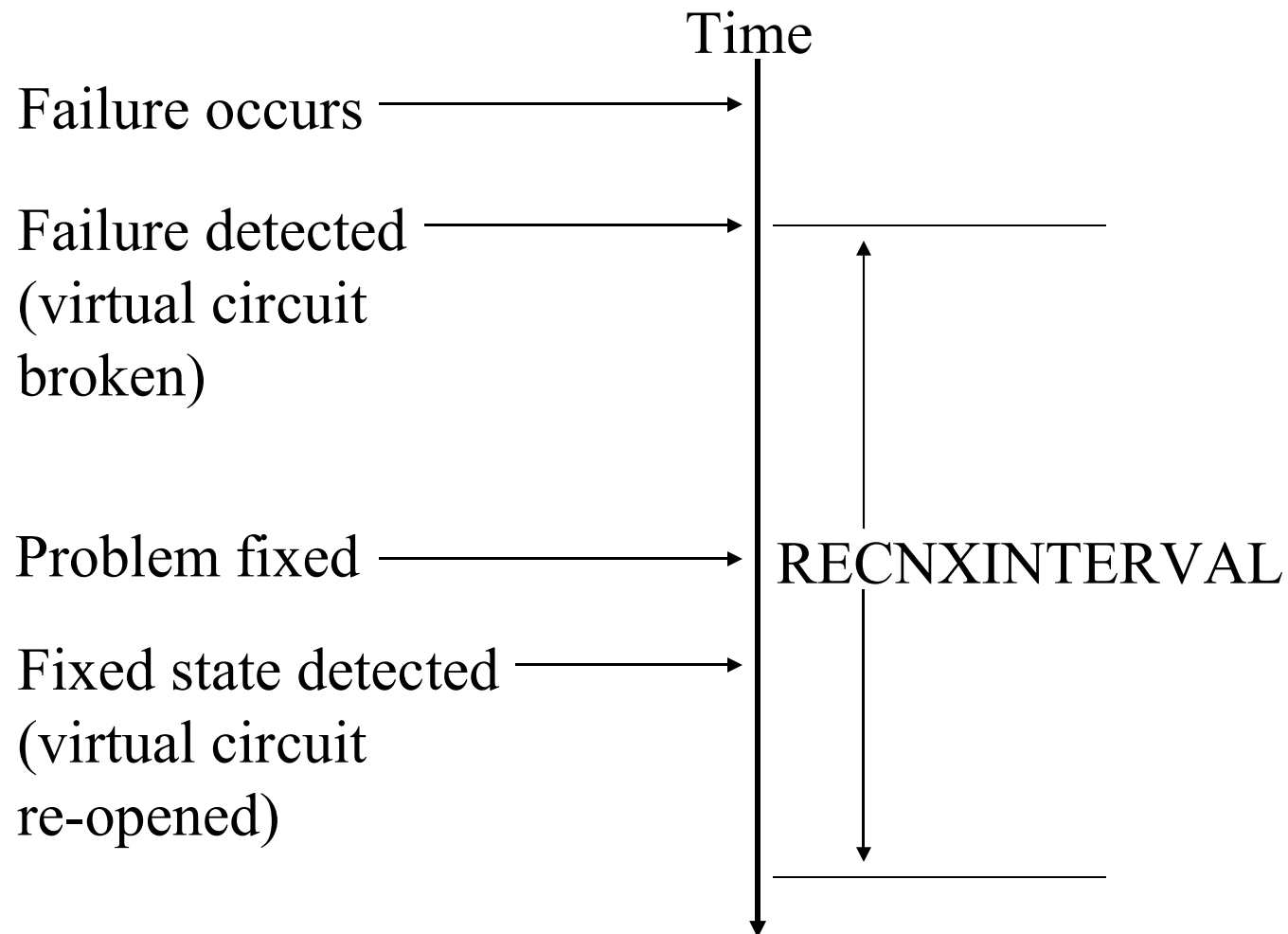
Failure Detection and Recovery Times



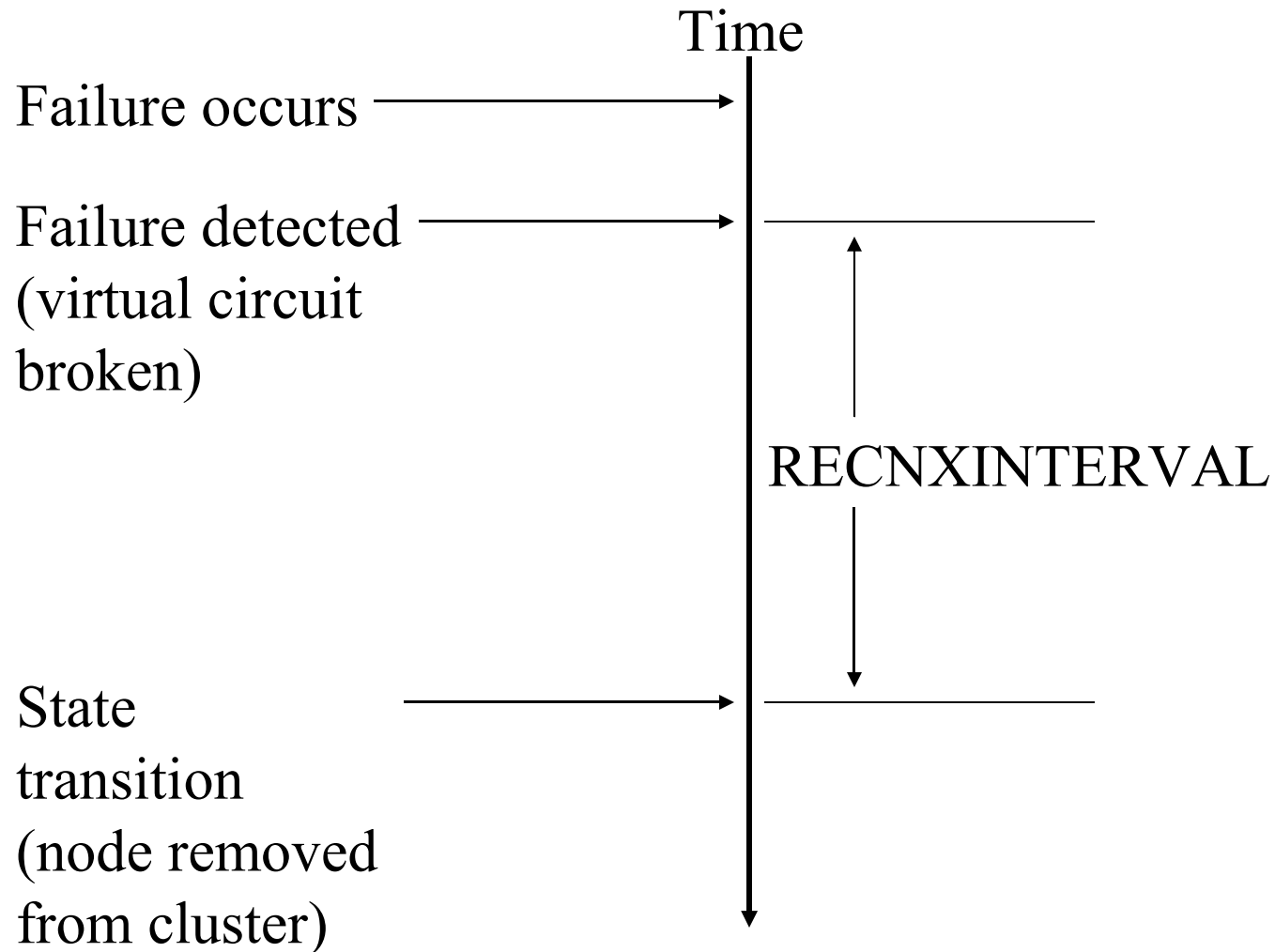
Portions of Failure and Recovery times

- Failure detection time
- Period of “patience”
- Failover or Recovery time

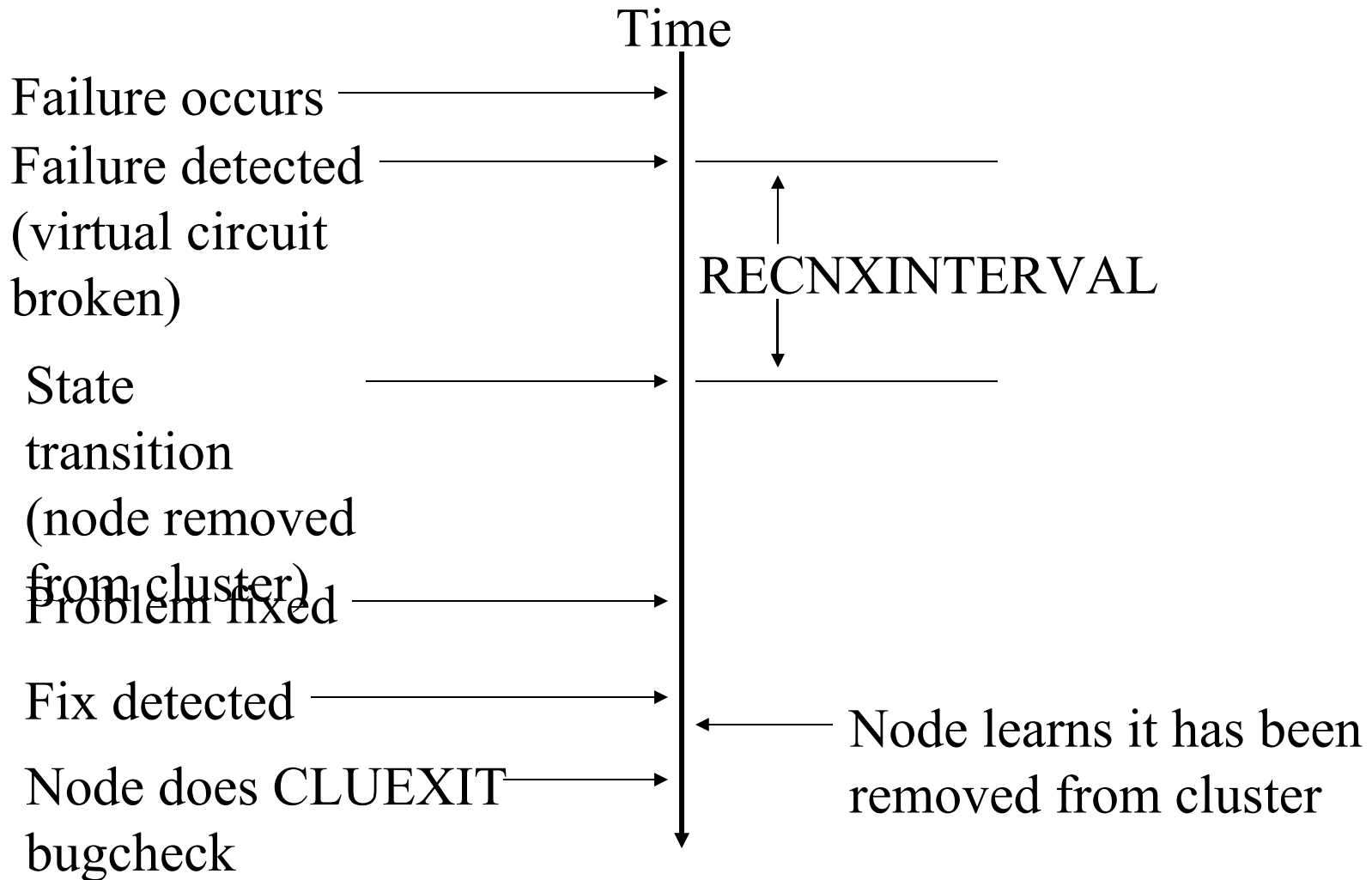
Failure and Repair/Recovery within Reconnection Interval



Hard Failure



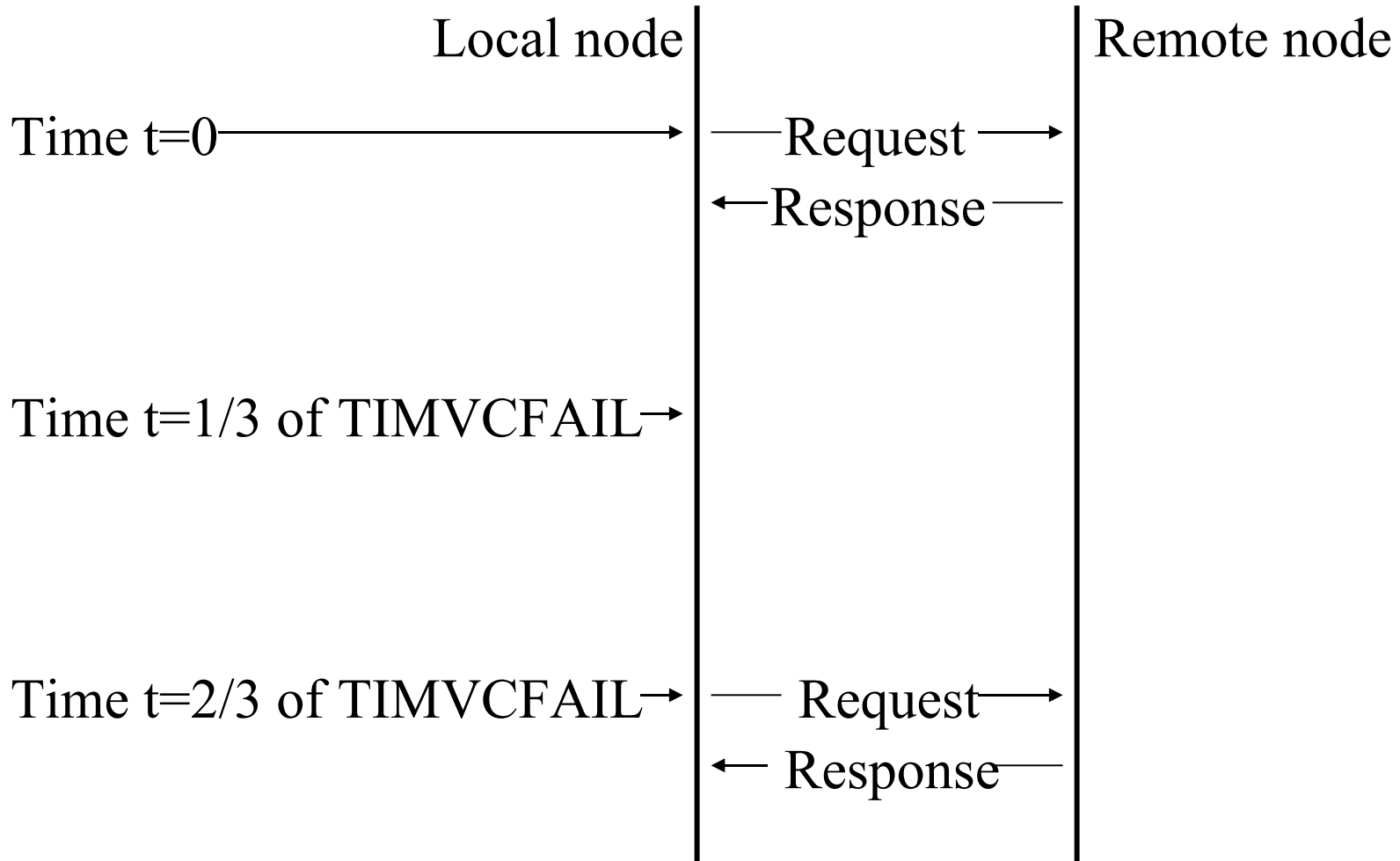
Late Recovery



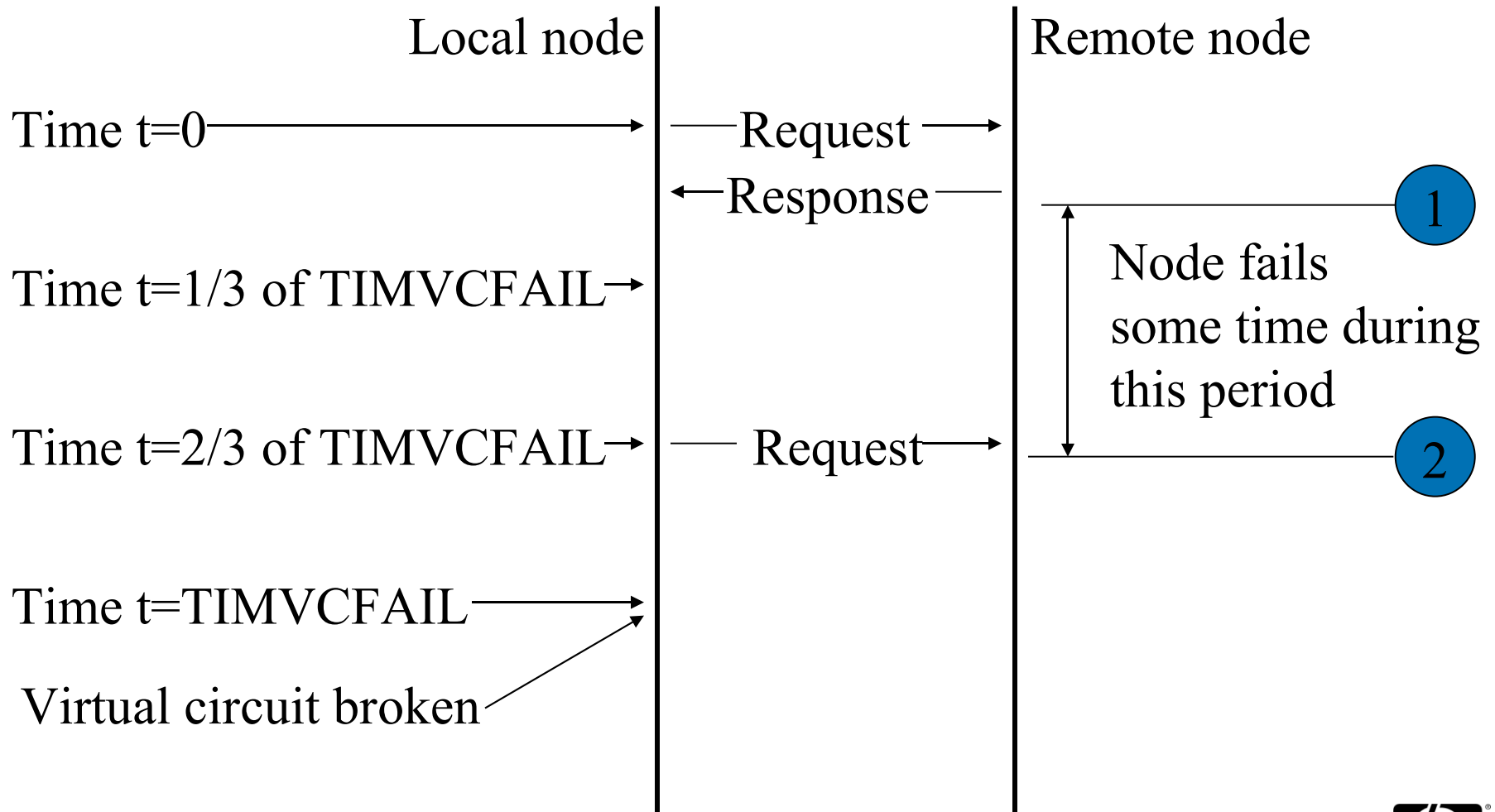
Failure Detection Mechanisms

- Mechanisms to detect a node or communications failure
 - Last-Gasp Datagram
 - Periodic checking
 - Multicast Hello packets on LANs
 - Polling on CI and DSSI
 - **TIMVCFAIL** check

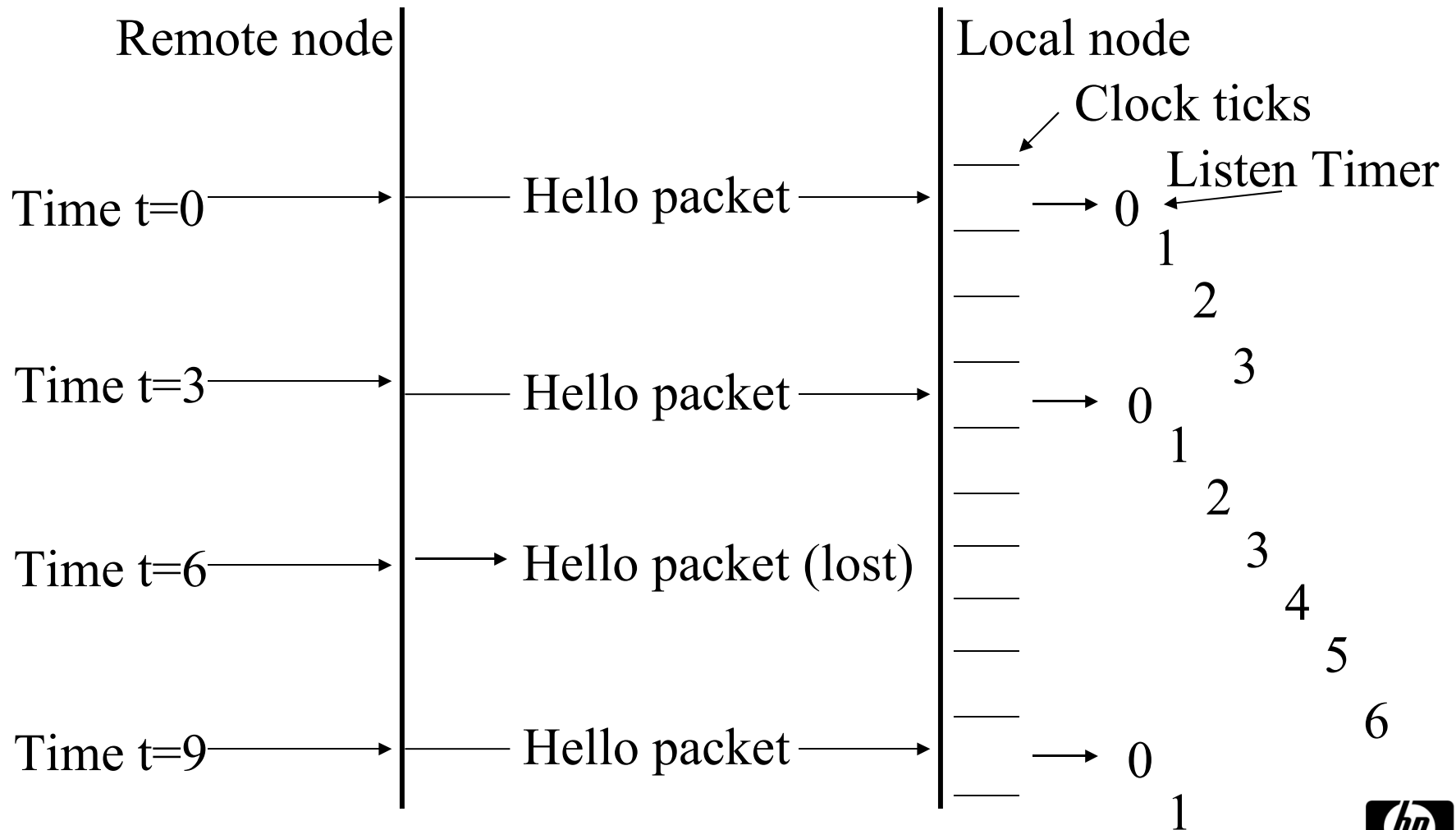
TIMVCFAIL Mechanism



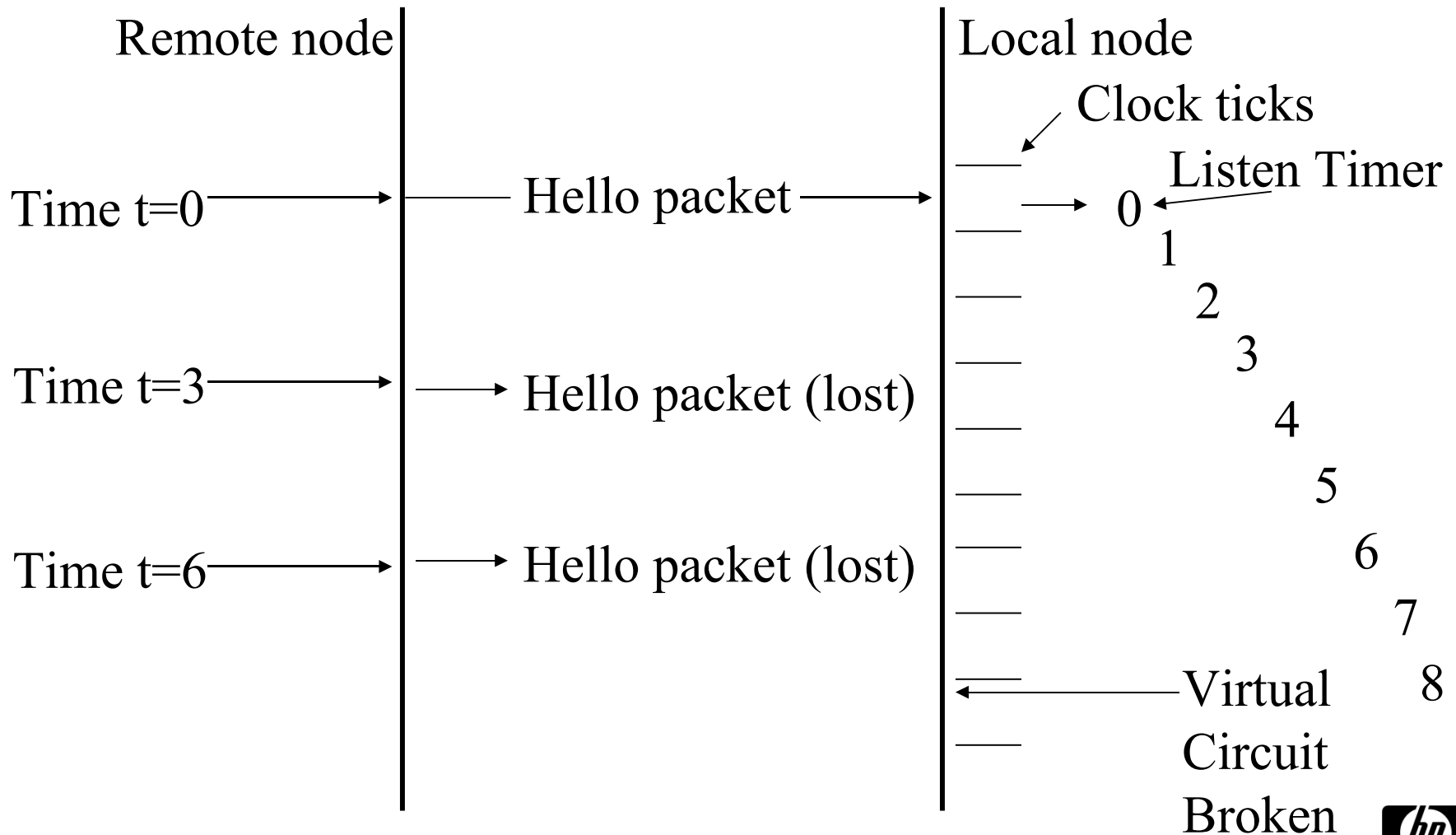
TIMVCFAIL Mechanism



Failure Detection on LAN interconnects



Failure Detection on LAN interconnects



Sequence of events During a State Transition

- Determine new cluster configuration
 - If quorum is lost:
 - QUORUM capability bit removed from all CPUs
 - no process can be scheduled to run
 - Disks all put into mount verification
 - If quorum is not lost, continue...
- Rebuild lock database
 - Stall lock requests
 - I/O synchronization
 - Do rebuild work
 - Resume lock handling

Measuring State Transition Effects

- Determine the type of the last lock rebuild:

```
$ ANALYZE/SYSTEM
```

```
SDA> READ SYS$LOADABLE_IMAGES:SCSDEF
```

```
SDA> EVALUATE @(@CLU$GL_CLUB + CLUB$B_NEWRBLD_REQ) & FF
```

```
Hex = 00000002    Decimal = 2    ACP$V_SWAPPRV
```

- Rebuild type values:
 1. Merge (locking not disabled)
 2. Partial
 3. Directory
 4. Full

Measuring State Transition Effects

- Determine the duration of the last lock request stall period:

```
SDA> DEFINE TOFF = @(@CLU$GL_CLUB+CLUB$L_TOFF)
```

```
SDA> DEFINE TON = @(@CLU$GL_CLUB+CLUB$L_TON)
```

```
SDA> EVALUATE TON-TOFF
```

```
Hex = 0000026B    Decimal = 619    PDT$Q_COMQH+00003
```

Minimizing the Impact of State Transitions



Avoiding State Transitions

- Provide redundancy for single points of failure
 - Multiple redundant cluster interconnects
 - OpenVMS can fail-over and fail-back between different interconnect types
 - Protect power with UPS, generator
 - Protect/isolate network if LAN is used as cluster interconnect
 - Minimize number of nodes which might fail and trigger a state transition

Minimizing Duration of State Transitions

- Configurations issues:
 - Few (e.g. exactly 3) nodes
 - Quorum node; no quorum disk
 - Isolate cluster interconnects from other traffic
- Operational issues:
 - Avoid disk rebuilds on reboot
 - Let Last-Gasp work

System Parameters which affect State Transition Impact: Failure Detection

- TIMVCFAIL can provide guaranteed worst-case failure detection time
 - Units of 1/100 second; minimum possible value 100 (1 second)
- When using LAN as the Cluster Interconnect:
 - LAN_FLAGS bit 12 (%x1000) enables Fast Transmit Timeout
 - PE4 parameter can control PEDRIVER Hello Interval and Listen Timeout values:
 - Longword, four 1-byte fields: <00> <00> HI> <LT>
 - <HI> is Hello packet transmit Interval in tenths of a second
 - Default 3 seconds (dithered)
 - <LT> is hello packet Listen Timeout value in seconds
 - Default 8 seconds
 - Listen Timeout < 4 seconds may have trouble communicating with boot driver
 - Ensure that TIMVCFAIL > Listen Timeout >= 1 second

System Parameters which affect State Transition Impact: Reconnection Interval

- Reconnection interval after failure detection:
 - RECNXINTERVAL
 - Units of 1 second; minimum possible value 1

Questions?

Speaker Contact Info:

- Keith Parris
- E-mail: Keith.Parris@hp.com **or** keithparris@yahoo.com
- Web: <http://www2.openvms.org/kparris/>



i n v e n t