



LAN Interconnect Monitoring for OpenVMS Clusters

Keith Parris
Multivendor Systems Engineering
HP Services



Topics



- OpenVMS and LAN redundancy
 - How OpenVMS detects LAN failures
- LAVC\$FAILURE_ANALYSIS facility
 - Theory of operation
 - Setup and use
 - Maintenance
- Available Freeware tools
 - EDIT_LAVC.COM
 - SIFT_LAVC.COM
 - LAVC\$FAILURE_OFF.MAR

OpenVMS and LAN Redundancy



- PEDRIVER is the code supporting SCS communications over LANs
- Since OpenVMS version 5.4-3, use of multiple LAN adapters for SCS has been supported
 - Allows cluster to continue operating despite LAN adapter, bridge, or cable failures
- While hardware redundancy allows one to survive a failure,
 - Failures must be detected and promptly fixed, or a subsequent second failure could cause an outage

PEDRIVER 'Hello' Packets



- Sent approximately every 1.5 to 3 seconds
 - “Dithered” to avoid packet “trains” forming
- PEDRIVER expects to receive Hello packets regularly on each possible path

Network Troubleshooting



- Locating the offending component in a network failure can be difficult
- OpenVMS provides a tool to make failure detection and failed-component identification easier in a LAVC environment: it's called the `LAVC$FAILURE_ANALYSIS` facility

Template Program



- Template program is found in `SYS$EXAMPLES:` and called `LAVC$FAILURE_ANALYSIS.MAR`
- Written in Macro-32
 - but you don't need to know Macro to use it
- Documented in Appendix D of OpenVMS Cluster Systems Manual
 - Appendix E (subroutines the above program calls) and Appendix F (general info on troubleshooting LAVC LAN problems) are also very helpful

Using LAVC\$FAILURE_ANALYSIS



- To use, the program must be:
 1. Edited to insert site-specific information
 2. Compiled (assembled on VAX)
 3. Linked, and
 4. Run at boot time on each node in the cluster

Maintaining LAVC\$FAILURE_ANALYSIS



- Program must be re-edited whenever:
 - The LAN used as a Cluster Interconnect is reconfigured
 - A node's MAC address changes
 - e.g. Field Service replaces a LAN adapter without swapping MAC address ROMs
 - A node is added or removed (permanently) from the cluster

How Failure Analysis is Done



- OpenVMS is told what the network configuration should be
- From this info, OpenVMS infers which LAN adapters should be able to “hear” Hello packets from which other LAN adapters
- By checking for receipt of Hello packets, OpenVMS can tell if a path is working or not

How Failure Analysis is Done



- By analyzing Hello packet receipt patterns and correlating them with a mathematical graph of the network, OpenVMS can tell what nodes of the network are passing Hello packets and which appear to be blocking Hello packets
- OpenVMS determines a Primary Suspect (and, if there is ambiguity as to exactly what has failed, an Alternate Suspect), and reports these via OPCOM messages with a “%LAVC” prefix

Getting Failures Fixed



- Since notification is via OPCOM messages, someone or something needs to be scanning OPCOM output and taking action
- ConsoleWorks, Console Manager, CLIM, or RoboCentral can scan for %LAVC messages and take appropriate action (e-mail, pager, etc.)

- Data required:
 - Local Area Network configuration:
 - OpenVMS Nodes
 - LAN adapters in each node
 - Bridges
 - Hubs
 - Links between all of the above

Network Information



- OpenVMS considers LAN building blocks as being divided into 4 classes:
 - **NODE**: The OpenVMS systems
 - **ADAPTER**: LAN host-bus adapters in each OpenVMS system
 - **COMPONENT**: Hubs, bridges, bridge-routers
 - **CLOUD**: Combinations of components that can't be diagnosed directly (more on this later)

Network building blocks



NODEs

OpenVMS
Node 1

OpenVMS
Node 2

Network building blocks



NODEs ADAPTERs

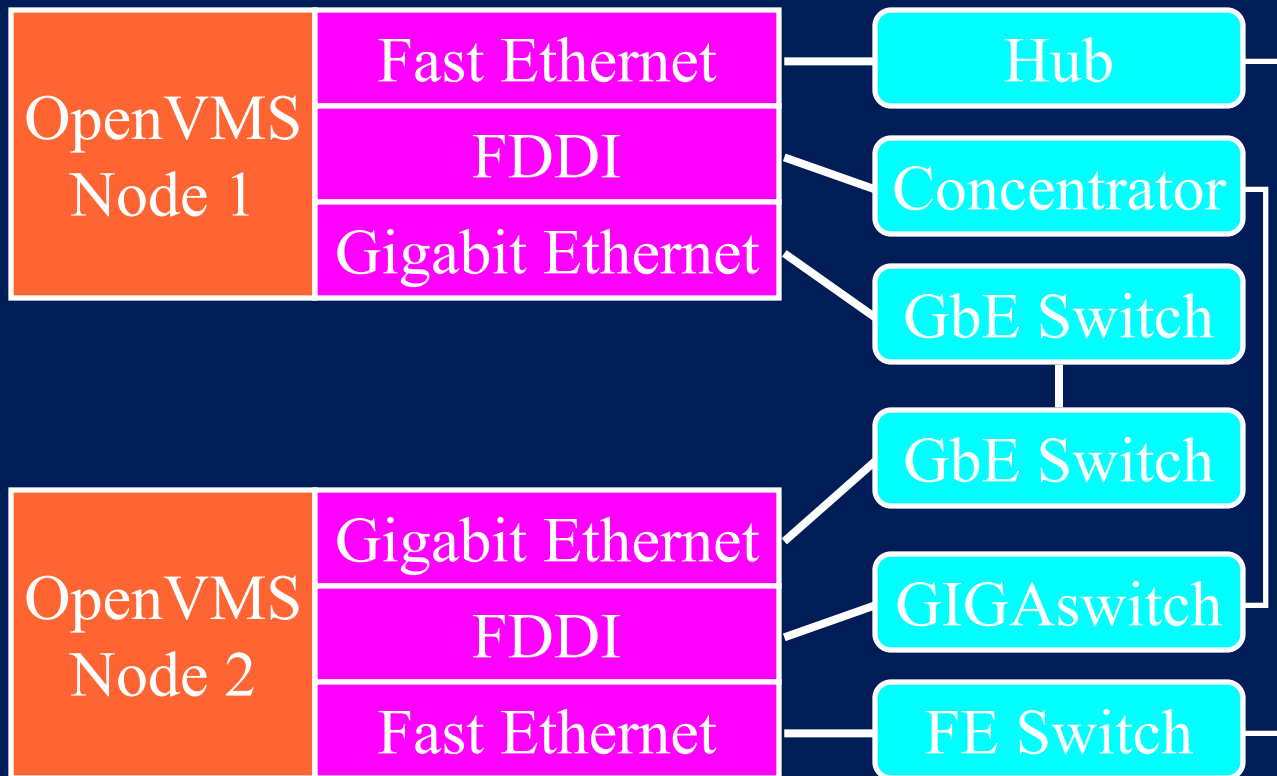
OpenVMS Node 1	Fast Ethernet
	FDDI
	Gigabit Ethernet

OpenVMS Node 2	Gigabit Ethernet
	FDDI
	Fast Ethernet

Network building blocks



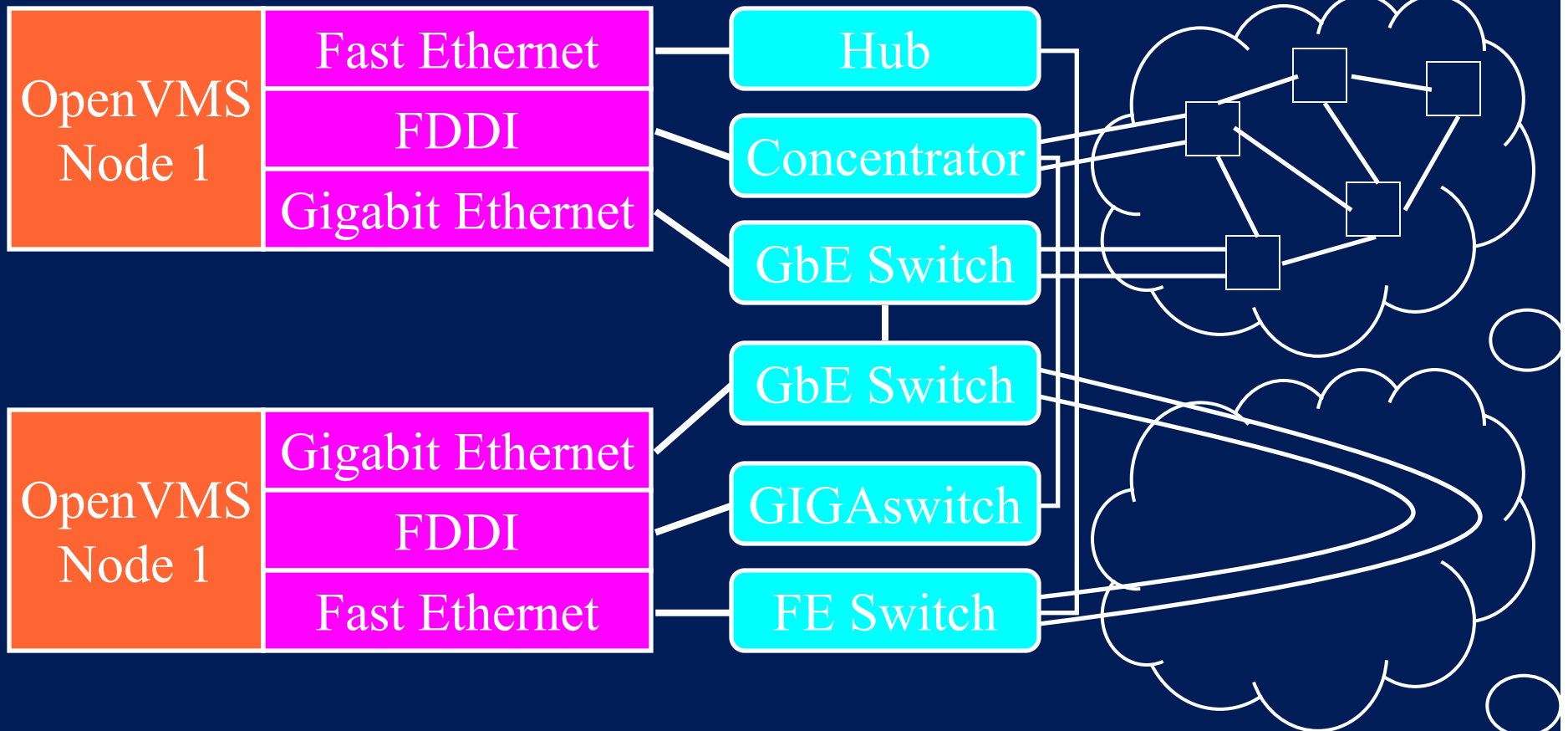
NODEs ADAPTERs COMPONENTs



Network building blocks



NODEs ADAPTERs COMPONENTs CLOUDs



Handling Network Loops

- The algorithm used for LAVC\$FAILURE_ANALYSIS can't deal with loops in the network graph
 - Yet redundancy is often configured among LAN components
 - The bridges' Spanning Tree algorithm shuts off backup links unless and until a failure occurs
 - Hello packets don't get through these backup links, so OpenVMS can't track them
 - For these cases, you replace the redundant portion of the network with a "network cloud" that includes all of the redundant components
 - Then OpenVMS can determine if the network "cloud" as a whole is functioning or not

Handling Redundancy



- Multiple, completely separate LANs don't count as "loops" and OpenVMS can track each one separately and simultaneously

Gathering Info



- Data required (more detail):
 - Node names and descriptions
 - LAN adapter types and descriptions, and:
 - MAC address
 - e.g. 08-00-2B-xx-xx-xx, 00-F8-00-xx-xx-xx
 - plus DECnet-style MAC address for Phase IV
 - e.g. AA-00-04-00-yy-zz

Getting MAC address info



```
$! SHOWLAN.COM
$!
$   write sys$output "Node ",f$getsi("nodename")
$   temp_file := showlan_temp.temp_file
$   call showlan/out='temp_file'
$   search 'temp_file' "(SCA)","Hardware Address"
_
           /out='temp_file`-1
$   delete 'temp_file';*
$   search/window=(0,1) 'temp_file`-1 "(SCA)"
$   delete 'temp_file`-1;*
$   exit
$!
$ showlan: subroutine
$   analyze/system
show lan/full
exit
$   endsubroutine
```

Editing the Program



- Once the data is gathered, you edit the program
- There are 5 sections to edit, as follows:

Edit 1



- In Edit 1, you can give descriptive names to Nodes, Adapters, Components, and Clouds
- These names become names of macros which you'll create invocations of later in the code

Edit 1



```
;      Edit 1.  
;  
;describe      Define the hardware components needed to  
;              the physical configuration.  
;
```

```
NEW_COMPONENT  SYSTEM          NODE  
NEW_COMPONENT  LAN_ADP         ADAPTER  
NEW_COMPONENT  DEMPR           COMPONENT  
NEW_COMPONENT  DELNI           COMPONENT  
NEW_COMPONENT  SEGMENT        COMPONENT  
NEW_COMPONENT  NET_CLOUD       CLOUD
```

Edit 2



- In Edit 2, you create “ASCII art” to document the LAVC LAN configuration
- This has no functional effect on the code, but helps you (and others who follow you) understand the information in the sections which follow
- In the drawing, you choose brief abbreviated names for each network building block (Node, Adapter, Component, or Cloud)
 - These abbreviated names are only used within the program, and do not appear externally

Edit 3



- In Edit 3, you name and provide a text description for each system and its LAN adapter(s), and the MAC address of each adapter
 - The name and text description will appear in OPCOM messages indicating when failure or repair has occurred
 - The MAC address is used to identify the origin of Hello messages

- For DECnet Phase IV, which changes the MAC address on all circuits it knows about from the default hardware address to a special DECnet address when it starts up, you provide both:
 - The hardware MAC address (e.g. 08-00-2B-nn-nn-nn) and
 - The DECnet-style MAC address which is derived from the DECnet address of the node (AA-00-04-00-yy-xx)
- DECnet Phase V does not change the MAC address, so only the HW address is needed

Edit 4



- In Edit 4, you name and provide a text description for each Component and each Cloud
 - The name and text description will appear in OPCOM messages indicating when failure or repair has occurred

Edit 4



```
;      Edit 4.
;
;      Label each of the other network components.
;

DEMPR   MPR_A, , <Connected to segment A; In the Computer room>
DELNI   LNI_A, , <Connected to segment B; In the Computer room>

SEGMENT Sa, , <Ethernet segment A>
SEGMENT Sb, , <Ethernet segment B>

NET_CLOUD      BRIDGES, , <Bridging between ethernet segments A and B>
```

Edit 5



- In Edit 5, you indicate which network building blocks have connections to each other
- This is a list of pairs of devices, indicating they are connected

Edit 5



```
;      Edit 5.
;
;      Describe the network connections.
;

CONNECTION      Sa,      MPR_A
CONNECTION      MPR_A,      A1
CONNECTION      A1,      A
CONNECTION      MPR_A,      B1
CONNECTION      B1,      B

CONNECTION      Sa,      D1
CONNECTION      D1,      D

CONNECTION      Sa,      BRIDGES
CONNECTION      Sb,      BRIDGES

CONNECTION      Sb,      LNI_A
CONNECTION      LNI_A,      A2
CONNECTION      A2,      A
CONNECTION      LNI_A,      B2
CONNECTION      B2,      B

CONNECTION      Sb,      D2
CONNECTION      D2,      D
```

EDIT_LAVC.COM Tool



A DCL command procedure is available to gather the information and create an example LAVC\$FAILURE_ANALYSIS.MAR program customized for a given cluster. See

- http://encompasserve.org/~parris/edit_lavc.com and EDIT_LAVC_DOC.TXT at the same location
- These are also on the V6 Freeware CD for OpenVMS under directory [KP_CLUSTERTOOLS]

To create a customized version of LAVC\$FAILURE_ANALYSIS.MAR and deposit it into your default directory, do:

```
$ @EDIT_LAVC
```

To compile/assemble and link the resulting program, do:

```
$ @EDIT_LAVC BUILD
```

OPCOM Messages Generated



- On a failure, LAVC\$FAILURE_ANALYSIS identifies at least one Primary Suspect:
 - **%LAVC-W-PSUSPECT**, *<device_description>*
- If there is more than one device whose failure might produce the same symptoms, LAVC\$FAILURE_ANALYSIS can also identify one or more Alternate Suspects:
 - **%LAVC-I-ASUSPECT**, *<device_description>*
- When the repair of a Suspect device (either Primary or Alternate) is detected, this is reported:
 - **%LAVC-S-WORKING**, *<device_description>*

Customization with EDIT_LAVC.COM



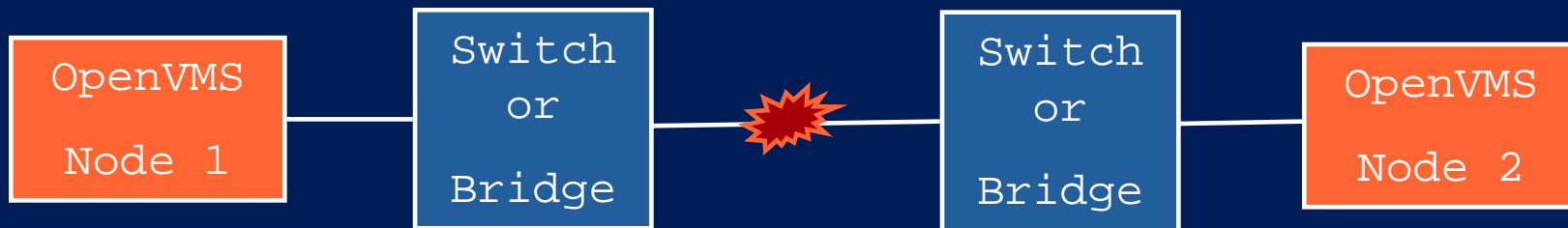
- EDIT_LAVC.COM tries to make up reasonable default descriptions for nodes, adapters, and network segments. You can override these with logical names. For example:

```
$!  
$! Create LAVC$FAILURE_ANALYSIS.MAR file customized for XYZ cluster  
$!  
$ DEFINE EDIT_LAVC_DESC_CLUSTER "XYZ"  
$ DEFINE EDIT_LAVC_DESC_NODE_ABC ", Alphaserver ES45 in XYZ cluster"  
$ DEFINE EDIT_LAVC_DESC_NODE_ABC_ADAPTER_EWA "Device EWA, Node ABC, DEGPA"  
$ DEFINE EDIT_LAVC_DESC_NODE_ABC_ADAPTER_EWB "Device EWB, Node ABC, DE500"  
$ DEFINE EDIT_LAVC_DESC_NODE_ABC_ADAPTER_FWA "Device FWA, Node ABC, DEFPA (left)"  
$ DEFINE EDIT_LAVC_DESC_NODE_ABC_ADAPTER_FWB "Device FWB, Node ABC, DEFPA (right)"  
.  
.  
.  
$ DEFINE EDIT_LAVC_DESC_VLAN_1 "Cisco VLAN 123 (Fast Ethernet); XYZ cluster"  
$ DEFINE EDIT_LAVC_DESC_VLAN_2 "GIGAswitch A (FDDI); XYZ cluster"  
$ DEFINE EDIT_LAVC_DESC_VLAN_3 "Cisco VLAN 456 (Gigabit Ethernet); XYZ cluster"  
$ DEFINE EDIT_LAVC_DESC_VLAN_4 "GIGAswitch B (FDDI); XYZ cluster"  
$ @EDIT_LAVC
```

Correlating Error Messages Between Nodes



- %LAVC OPCOM messages from each node show the failure *from the viewpoint of that specific node*
 - You can often get a better feel for the actual underlying failure by comparing the failure messages as reported from each node



SIFT_LAVC.COM Tool



•A DCL command procedure is available to gather all %LAVC messages from OPERATOR.LOG files and sort them in timestamp order to allow easier correlation of the events from the viewpoint of each node. See:

http://encompasserve.org/~parris/sift_lavc.com or the V6 Freeware CD for OpenVMS

To summarize %LAVC messages from the current (highest-numbered) version of OPERATOR.LOG files on all nodes, do:

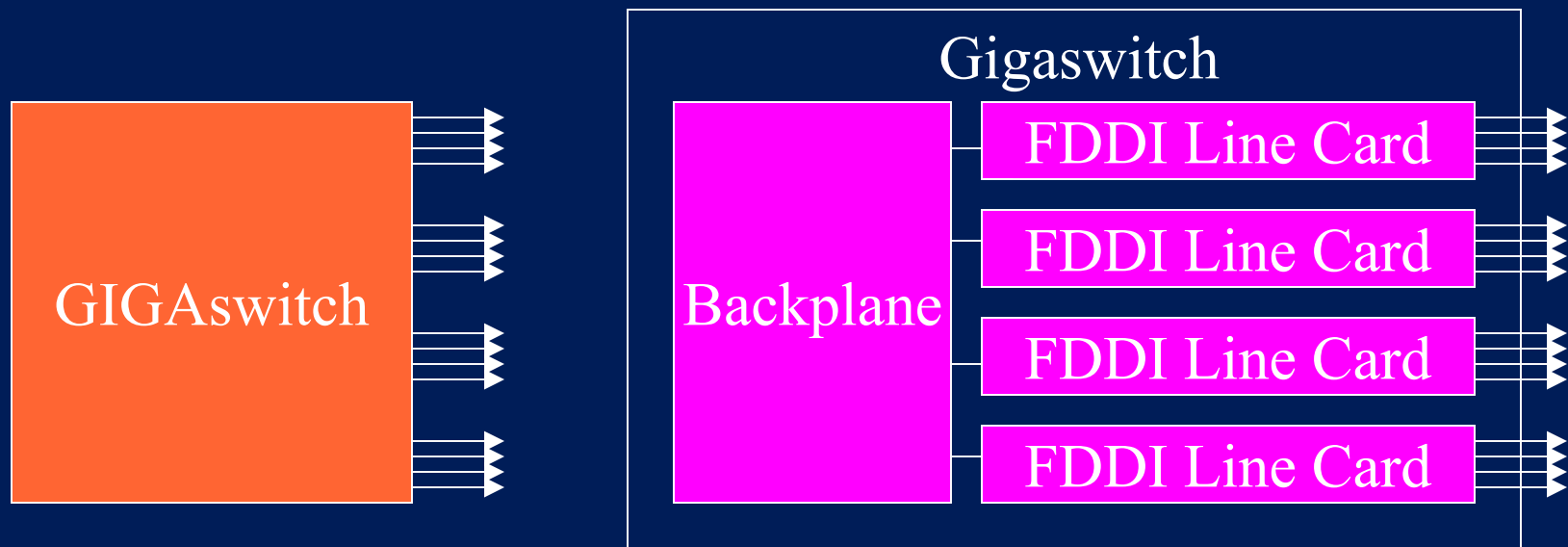
```
$ @SIFT_LAVC
```

Level of Detail



- There is a trade-off between level of detail in diagnostic info and the amount of work required to initially set up and to maintain the program over time
 - More detail means more work to setup, and more maintenance work, but can provide more-specific diagnostic info when failures occur

Level of Detail Example



Disabling LAVC\$FAILURE_ANALYSIS



To turn off LAVC Failure Analysis, use the LAVC\$FAILURE_OFF.MAR program found at <http://encompasserve.org/~parris/> or on the V6 Freeware CD for OpenVMS

Speaker Contact Info



- Keith Parris
- E-mail: Keith.Parris@hp.com or parris@encompasserve.org or keithparris@yahoo.com
- Web: <http://encompasserve.org/~kparris/> and <http://www2.openvms.org/kparris/> and <http://www.geocities.com/keithparris/>



i n v e n t