HEALTHMON

DU-06718-001 _vR340 | July 2014

Best Practices and User Guide

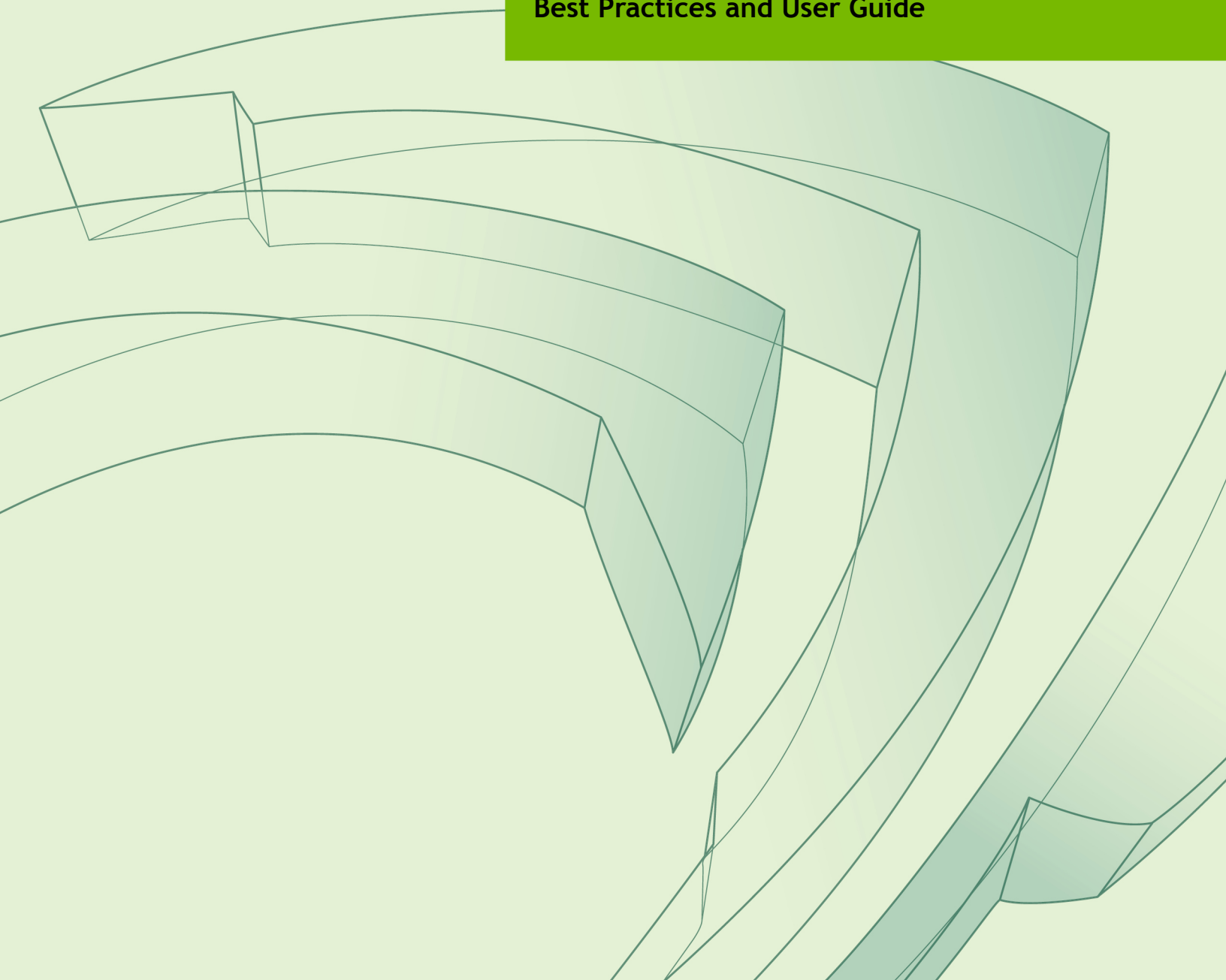

TABLE OF CONTENTS

LIST OF TABLES

Chapter 1. OVERVIEW

nvidia-healthmon is the system administrator's and cluster manager's tool for detecting and troubleshooting common problems affecting NVIDIA® Tesla™ GPUs in a high performance computing environment. nvidia-healthmon contains limited hardware diagnostic capabilities, and focuses on software and system configuration issues.

1.1. nvidia-healthmon Goals

nvidia-healthmon is designed to:

1. Discover common problems that affect a GPUs ability to run a compute job including:
   - Software configuration issues
   - System configuration issues
   - System assembly issues, like loose cables
   - A limited number of hardware issues
2. Provide troubleshooting help
3. Easily integrate into *Cluster Scheduler* and *Cluster Management* applications
4. Reduce downtime and failed GPU jobs

1.1.1. Beyond the Scope of nvidia-healthmon

nvidia-healthmon is not designed to:

1. Provide comprehensive hardware diagnostics
2. Actively fix problems

1.2. Dependencies

This version of nvidia-healthmon depends on the NVIDIA Developer Display Driver r340, or later, found on the CUDA download page. http://developer.nvidia.com/cuda-downloads

1.3. Supported Products

nvidia-healthmon supports Tesla GPUs running on Linux (bare metal) operating systems. NVIDIA® Tesla™ Line:

- All Fermi architecture GPUs
- All Kepler architecture GPUs

Chapter 2. RUNNING NVIDIA-HEALTHMON

Once unpackaged, nvidia-healthmon can be run from the command line.

nvidia-healthmon will create a CUDA context on the GPU it is testing. In compute exclusive mode, this means that no other CUDA process will be able to create a CUDA context on the GPU. The results of nvidia-healthmon may also be affected by other CUDA processes. For example, nvidia-healthmon may detect low bandwidth if another process is sending data over PCIe to any GPU on the system.

```
user@hostname
$ nvidia-healthmon
```

When no arguments are supplied, nvidia-healthmon will run with the default behavior on all supported GPUs.

2.1. Listing GPUs

nvidia-healthmon is able to list the GPUs installed on the system. This is useful to determine the PCI bus ID or device index needed in the next section.

The device index may change if the NVIDIA display driver is reloaded, or the system is rebooted.

```
user@hostname
$ nvidia-healthmon -L
```

For extended GPU information see the nvidia-smi tool:

```
user@hostname
$ man nvidia-smi

user@hostname
$ nvidia-smi -q
```

2.2. Targeting a Specific GPU

nvidia-healthmon can target a single GPU or a set of GPU's. To target a specific GPU, run nvidia-healthmon using the **-i** or **--id** flag with the identifier of the GPU to be targeted. Identifiers are either:

1. A device index
2. A PCI bus ID
3. A GPU chip UUID
4. A GPU board serial number

A board serial number will target all GPUs on the board.

```
user@hostname
$ nvidia-healthmon -i 0

user@hostname
$ nvidia-healthmon -i 0000:02:00.0
```

2.3. Modes

The default mode is quick mode. In quick mode a subset of tests are run, to quickly detect common problems. The **-q** or **--quick** flags allow quick mode to be explicitly requested.

The other mode is extended mode. In extended mode all available tests will be run.

To run extended diagnostics run nvidia-healthmon using the **-e** or **--extended** flags.

```
user@hostname
$ nvidia-healthmon --extended
```

For more information about these modes see the *nvidia-healthmon Best Practices Guide*.

2.4. Log File Output

By default nvidia-healthmon will report information to standard output. To redirect output to a file, the **-l** or **--log-file** flags are used. Only errors in command line parsing will be printed to the console.

2.5. Verbose Output

The default output of nvidia-healthmon will not report values for various metrics it has collected. The verbose flags, **-v** or **--verbose**, can be used to print values like the pinned memory bandwidth and CUDA device query information. Additionally, verbose mode will provide information about why tests were skipped.

The output format of nvidia-healthmon may change in a later release.

2.6. Other Flags

For a more complete list of the flags available, run nvidia-healthmon with the **-h/--help** or **-H/--verbose-help** flags.

```
user@hostname
$ nvidia-healthmon -h
```

Chapter 3. INTERPRETING THE RESULT

3.1. nvidia-healthmon Exit Code

nvidia-healthmon will terminate once it completes the execution diagnostics on all specified devices. An exit code of **zero** will be used when nvidia-healthmon runs successfully. A **non-zero** exit code indicates that there was a problem with the nvidia-healthmon run. This may be due to problems running the diagnostic: such as a missing configuration file, or invalid command line arguments, or problems with the local machine that nvidia-healthmon has detected. The output of the application must be read to determine what the exact problem was.

An example of a successful run of nvidia-healthmon.

```
user@hostname
$ nvidia-healthmon -q

Loading Config:  SUCCESS
Global Tests
   NVML Sanity: SUCCESS
   Tesla Devices Count: SUCCESS
   Global Test Results:  2  success, 0  errors, 0  warnings, 0  did not run
-------------------------------------------------------------
GPU  0000:04:00.0 #1  : Tesla C2075  (Serial: 0425912072221)
   NVML Sanity: SUCCESS InfoROM: SKIPPED
   GEMINI  InfoROM: SKIPPED
   ECC:  SUCCESS
   CUDA  Sanity
      Result: SUCCESS
   PCIe  Maximum Link Generation: SUCCESS
   PCIe  Maximum Link Width: SUCCESS
   PCI  Seating: SUCCESS
   PCI  Bandwidth: SKIPPED
   Device Results:  6  success, 0  errors, 0  warnings, 3  did not run

System  Results: 8  success, 0  errors, 0  warnings, 3  did not run
```

In the above example three of the tests did not run.

- The InfoROM test is an extended mode only test so it is skipped. To run the InfoROM test, extended mode must be used.
- The Gemini infoROM test was skipped because the Tesla C2075 only has a single GPU on the board.
- The PCIe bandwidth test was not run because the configuration file does not specify the bandwidth supported by the system. To run the bandwidth test, the configuration file must be edited.

Skipped tests will not affect the nvidia-healthmon exit code.

An example of a failing run of nvidia-healthmon.

```
user@hostname
$ nvidia-healthmon -q

Loading Config: SUCCESS
Global Tests
   NVML Sanity: SUCCESS
   Tesla Devices Count: SUCCESS
   Global Test Results: 2 success, 0 errors, 0 warnings, 0 did not run
------------------------------------------------------------
GPU  0000:04:00.0 #1: Tesla C2075 (Serial: 0425912072221)
   NVML Sanity: SUCCESS
   InfoROM: SKIPPED
   GEMINI InfoROM: SKIPPED
   ECC: SUCCESS
   CUDA Sanity
      Result: SUCCESS
   PCIe Maximum Link Generation: SUCCESS
   PCIe Maximum Link Width: SUCCESS
   PCI Seating
      ERROR: After enabling maximum performance mode, the current PCIe Link
      Width (8) does not match the expected maximum PCIe Link Width (16). This
      can indicate that this GPU is improperly seated.
      An issue was detected with this GPU. This issue is usually caused by poor
      connection between the GPU and the system.
      * Run 'nvidia-smi -q'. In some cases this will report a poorly connected
        power cable.
      * Power down your system.
      * Check that all power   connectors are firmly attached (some GPUs require
        two power cables attached).
         * Check that the power cable is not damaged. Symptoms of damaged power
           cables include exposed wiring and kinks (sharply creased region).
         * Check that the power cable is attached to a working power supply.
      * Rerun these diagnostics, using the same GPU, on system that is known to
        be working. A variety of system issues can cause diagnostic failure.
      * Restart your system and install the latest NVIDIA display driver.
      * Contact your OEM  provider, to run further system diagnostics.
         * Run 'nvidia-bug-report.sh' as the root user.
         * Run 'nvidia-healthmon -v -e -l nvidia-healthmon-report.txt --debug'
         * Provide the files  nvidia-bug-report.log.gz,
           nvidia-healthmon-report.dump, and nvidia-healthmon-report.txt to
           your OEM to assist your OEM in resolving this issue.
      Result: ERROR
  PCI  Bandwidth: SKIPPED
  Device Results:  5 success, 1 errors, 0 warnings, 3 did not run

System  Results: 7  success, 1  errors, 0  warnings, 3  did not run
WARNING: One or more tests didn't run. Read the output for details.
ERROR: One or more tests failed. Read the output for details.
```

In the above example, nvidia-healthmon detected a problem with how the GPU was inserted into the system. nvidia-healthmon exited with a **non-zero** exit code. Additionally, the output provides a user readable description of what went wrong and a list of the steps the customer can take to solve the problem.

Chapter 4. CONFIGURING NVIDIA-HEALTHMON

While nvidia-healthmon will work out of the box without additional configuration, it is possible to configure the behavior and enable optional features. nvidia-healthmon is packaged with a sample configuration file, **nvidia-healthmon.conf**. This configuration file can be used to enable optional tests. By default, a test that is missing configuration information will be skipped.

The configuration file used can be specified on the command line. When not specified nvidia-healthmon will check the NVHEALTHMON_CONF environment variable for the full path (including file name) of the **nvidia-healthmon.conf** file, followed by the current working directory and lastly, the default RPM/DEB installation directory (**/etc/nvidia-healthmon/nvidia-healthmon.conf**).

Specify the configuration file using the following:

```
user@hostname
$ nvidia-healthmon -c /path/to/your/nvidia-healthmon.conf
```

A default **nvidia-healthmon.conf** file is provided with nvidia-healthmon. All of the available properties are listed in this file, with their descriptions.

4.1. Configuration File Contents

This file is formatted in the standard ini file format.

There are two types of sections in the file. The first is the **global** section, the second is a **GPU name** section. The global section contains expected system configuration information. For example, the **devices.tesla.count** describes the number of GPUs on the system. Any section name other than **global** is a GPU name section. The GPU name section contains information about all GPUs with a given name. For example, the section [Tesla M2090] contains the configuration for all Tesla M2090 GPUs. The two types of sections support different key value pairs. For instance, the key **devices.tesla.count** is only allowed to be grouped underneath the **global** category.

The **nvidia-healthmon.conf** file nvidia-healthmon is packaged with contains the valid key value pairs for each section along with a brief description.

Sample System Configuration File with 4 Devices shows a sample configuration file for a system that contains 4 devices that are either Tesla™ C2075s, or Tesla™ C2070s.

Sample System Configuration File with 4 Devices

```
[global]
devices.tesla.count = 4

[Tesla C2075]
bandwidth.warn = 1500
bandwidth.min = 100
pci.gen = 2
pci.width = 16
temperature.warn = 95

[Tesla C2070]
bandwidth.warn = 1500
bandwidth.min = 100
pci.gen = 2
pci.width = 16
temperature.warn = 90
```

4.2. Summary of valid keys

Table 1 Global keys

Key	Description
devices.tesla.count	Enable this setting to ensure that the expected number of Tesla brand GPUs are detected by the NVML library. This count only includes Tesla brand GPUs that the nvidia-healthmon process has sufficient permission to access. If this setting is not configured, then checks that require it will skip.
drivers.blacklist	Checks the system for drivers that have been known to cause issues with NVIDIA hardware, drivers, and software. If nvidia-healthmon detects any blacklisted drivers it will not execute further tests. You may add/remove drivers on this list at your own risk. If this setting is not configured, then checks that require it will skip.
display.gpudirect	Enable this setting to display the GPUDirect communication matrix between all P2P/RDMA devices on the system.
version.driver	Set this to give an expected Nvidia driver version, a mismatch causes the test to fail
rdma.enable	Set this to activate RDMA testing features and display relevant RDMA information in the GPU Direct communication matrix (through **display.gpudirect**). Please see **rdma.*** below

Table 2 SKU keys

Key	Description
bandwidth.warn	If the bandwidth from the host to GPU or from the GPU to the host is below this value (in MB/s), nvidia-healthmon will generate a warning. If this setting is not configured, then checks that require it will skip are detected by the NVML library.
bandwidth.min	If the bandwidth from the host to GPU or from the GPU to the host is below this value (in MB/s), nvidia-healthmon will generate an error. If this setting is not configured, then checks that require it will skip.
peer.bandwidth.warn	In the case that peer access is supported, if the bandwidth from one GPU to the other GPU is supported is below this value (in MB/s), nvidia-healthmon will generate a warning. If peer to peer access is not supported, the bandwidth test is still run, but no comparison to the minimum bandwidth is done.
peer.bandwidth.min	In the case that peer access is supported, if the bandwidth from one GPU to the other GPU is supported is below this value (in MB/s), nvidia-healthmon will generate an error. If peer to peer access is not supported, the bandwidth test is still run, but no comparison to the minimum bandwidth is done.
pci.gen	Compare the maximum PCIe link generation for the PCIe link closest to the GPU chip against the value specified here. If this setting is not configured, then checks that require it will skip. An error will be generated if there is a mismatch. For a board that contains multiple GPU chips, this value will reflect the PCIe link generation between the GPU chip and an on board PCIe switch. For single GPU boards this value reflects the link width between the GPU chip and the PCIe slot the GPU is connected to. Note that additional PCIe links upstream from the GPU may have a different link generation. Those links are not considered here. Because PCIe link generation is dependent on the non-GPU side of the link, knowledge of the system's capability is required to set the correct expectations. Consequently this config file disables this test for each GPU by default.
pci.width	Compare the maximum PCIe link width for the PCIe link closest to the GPU chip against the value specified here. If this setting is not configured, then checks that require it will skip. An error will be generated if there is a mismatch. For a board that contains multiple GPU chips, this value will reflect the PCIe link generation between the GPU chip and an on board PCIe switch. For single GPU boards this value reflects the link width between the GPU chip and the PCIe slot the GPU is connected to. Note that additional PCIe links upstream from the GPU may have a different link generation. Those links are not considered here. Because PCIe link generation is dependent on the non-GPU side of the link, knowledge of the system's capability is required to set the correct expectations. Consequently this config file disables this test for each GPU by default.

Key	Description
temperature.warn	Compare the current GPU die temperature to a warning level in degrees Celsius. A warning will be generated if the current temperature is at or above the warning level. Note that the desired temperature may vary based on the cooling system used. If this setting is not configured, then checks that require it will skip.
rdma.mlxdev rdma.mlxport	Perform an RDMA test to measure the bandwidth between a Mellanox Infiniband NIC and a GPU. The tests are activated via the rdma.enable key in the global section. rdma.mlxdev is the Mellanox device ID given by ibstat. rdma.mlxport is the port number you wish to perform the test on. Though this is strictly a loopback test, the port must be active and link up for the test to work correctly.
rdma.bandwidth.warn	In the case that RDMA access is supported, if the RDMA unidirectional bandwidth is below this value (in MB/s), nvidia-healthmon will generate a warning. RDMA bandwidths will be highly dependent on the exact host bridge version and architecture. This is an optional parameter.
rdma.bandwidth.min	In the case that RDMA access is supported, if the RDMA unidirectional bandwidth is below this value (in MB/s), nvidia-healthmon will generate an error. RDMA bandwidths will be highly dependent on the exact host bridge version and architecture. This is an optional parameter.
version.vbios	Check to make sure a device's vBios version is an expected value. This can be a single value or a comma-separated list of values.
ecc.check_state	Set this to check the ECC state of a device and consider a mismatch with this parameter to be a failure. If this parameter is provided, it must be one of the following: 0 = expect ECC to be disabled on this device 1 = expect ECC to be enabled on this device If this parameter is provided and a device does not support getting its ECC state, it will be considered a test failure

Chapter 5. GPUDIRECT

5.1. GPUDirect Testing

nvidia-healthmon can assist in the testing and optimization of GPUDirect communications. Enable the GPUDirect communication matrix using the **`display.gpudirect`** global key. Peer-to-peer (P2P) tests will automatically be conducted if there exists more than one GPUDirect-capable card in the system.

Please note, the enumeration of GPUs in the matrix are according to NVML.

In addition, RDMA tests can be conducted in conjunction with a Mellanox Infiniband card. The following conditions must be met for the RDMA test to correctly execute:

- The Infiband driver must be loaded and the IB link up
- The nv_peer_mem driver must be loaded. This driver and instructions for compiling it can be found on the Mellanox OFED page. http://www.mellanox.com/page/products_dyn?product_family=116
- The OFED libibverbs libraries installed and in the current library search path (i.e. LD_LIBRARY_PATH)

Enable the RDMA test using the global key **`rdma.enable`** and then in the appropriate SKU area use the keys **`rdma.mlxdev`** and **`rdma.mlxport`** to specify the IB card you wish to perform a GPUDirect RDMA test against. In addition, the **`display.gpudirectnic`** key can be used to include the IB card as part of the GPUDirect communication matrix described above.

Chapter 6. USE CASES

While nvidia-healthmon is primarily targeted at clusters of NVIDIA® Tesla™ GPUs, it can also be used in workstations without a cluster manager.

6.1. Run nvidia-healthmon after System Provisioning

After a system is provisioned, nvidia-healthmon can be run on the node to ensure that the node is correctly configured and able to run a GPU job. In this use case an extended mode run of nvidia-healthmon will try to deliver the most comprehensive system health check.

6.2. Run nvidia-healthmon before a Job

nvidia-healthmon can be run in a prologue or epilogue script in quick mode to perform a sanity check of the system and GPU. If nvidia-healthmon detects a problem the scheduler can mark the current node as down, and run the job on a different node to avoid job failure. In quick mode a subset of tests are run, such that a quick sanity test of the system is performed.

nvidia-healthmon will create a CUDA context on the device it is testing so it is often undesirable to run nvidia-healthmon when other processes are using the GPU to prevent unexpected behavior of either process.

6.3. Periodic Health Check

Analogous to periodically scanning for viruses, nvidia-healthmon can be run in the extended mode periodically. Again, when nvidia-healthmon reports a problem the scheduler can mark the node as down.

6.4. After Job Failure

When a GPU job fails, the extended mode run of nvidia-healthmon can help troubleshoot the problem.

6.5. Interfacing with Other Services

nvidia-healthmon can be run by a wrapper script which handles any reported warnings and errors. These problems can subsequently be forwarded to other services in order to notify them of any problems. NVIDIA suggests using syslog for logging reported issues on the system. Similarly, an SNMP trap can be sent to notify hosts over a network of these issues.

6.6. Troubleshooting Problems

nvidia-healthmon 's troubleshooting report is designed to cover common problems, and will often suggest a number of possible solutions. These troubleshooting steps should be tackled from the top down, as the most likely solution is listed at the top.

6.6.1. Save Log Files

NVIDIA recommends that the log files from failing nvidia-healthmon runs should be saved. Saving log files ensures that data about intermittent problems is not lost.

Some log files are encrypted and can only be decoded by NVIDIA engineers. These files are not corrupt. These logs contain a trace of the nvidia-healthmon run, and do not contain any sensitive information.