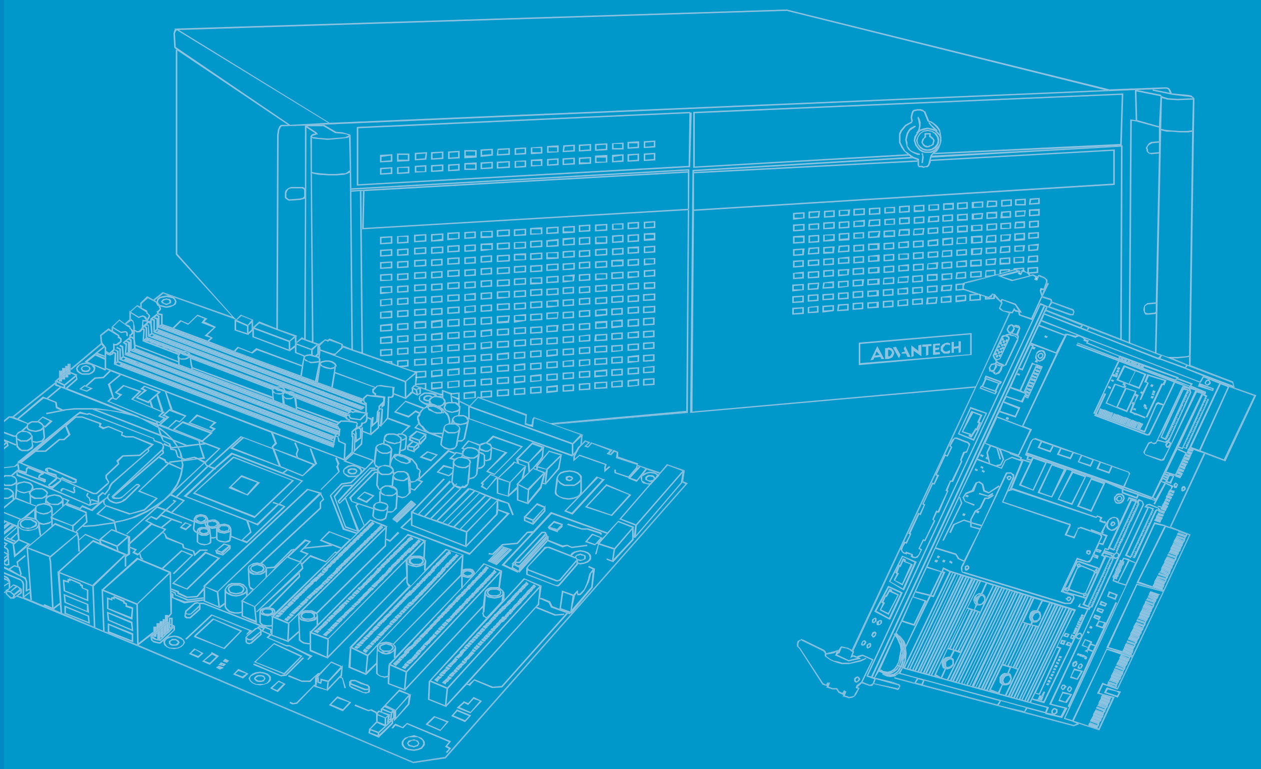


# SERVER IMANAGER - HEALTH GUARD

# USER MANUAL

**REVISION 6.0** **DATE 2020/10/30**

Enabling an Intelligent Planet

## Revision History

| Date [mm/dd/yyyy] | Revision | Modifications |
|---|---|---|
| 10/03/2018 | 0.1 | Initial version based on software package version 0.08 |
| 11/13/2018 | 1.0 | Finalize formatting and Release |
| 11/23/2018 | 1.1 | Removed DUI |
| 8/20/2019 | 1.2 | Added Section 6 Health Guard Exporter supporting Prometheus |
| 8/28/2019 | 2.0 | Edition 2 official release |
| 11/05/2019 | 2.1 | Removed lcd4linux<br>Modified “Metrics” sub-section<br>Added note for mcelog |
| 11/25/2019 | 2.2 | Modified the “Health Guard Exporter” sub-section based on software package version 0.06 |
| 12/5/2019 | 2.3 | Merged Health guard exporter manual |
| 12/24/2019 | 2.4 | Added the restart command for CLI |
| 1/2/2020 | 2.5 | Improved network threshold support |
| 1/3/2020 | 3.0 | Edition 3 official release |
| 2/12/2020 | 3.1 | Added storage and PCI for Health Guard Exporter |
| 3/19/2020 | 3.2 | Modified the “Health Guard Exporter” sub-section based on software package version 0.12 |
| 3/27/2020 | 4.0 | Edition 4 official release |
| 6/15/2020 | 4.1 | Modified the “Health Guard Exporter” sub-section. |
| 6/24/2020 | 5.0 | Edition 5 official release |
| 7/24/2020 | 5.1 | Added hostname information in “Health Guard Exporter” metrics. |
| 9/18/2020 | 5.2 | Modified the Installation sub-section. |
| 9/8/2020 | 6.0 | Edition 6 official release |

## About This Manual

The target audience of this manual includes users, developers, and technicians. This document describes the features, functions, and operations of Server iManager Health Guard and Exporter

This manual is organized as follows:

- Section 1 Outlines the steps for installing Health Guard
- Section 2 Introduces the CLI
- Section 3 Introduces the collector and events
- Section 4 Describes the failsafe and notification mechanisms
- Section 5 Provides information on operation and troubleshooting
- Section 6 Introduces the Health Guard Exporter and Prometheus stack
- The appendices provide supplemental information referenced in the other sections of this manual

This document covers:

- Server iManager Health Guard Exporter version v0.01 and later
- Server iManager Health Guard Exporter is supported with Health Guard version 0.12 and later

## Useful Documents

If you cannot find the information you are looking for or need more detailed information on a specific topic, please refer to the following documents and information sources. Should you require any assistance with obtaining these documents or if still cannot find what you are looking for, please contact your Advantech representative.

- An introduction of smartmontools can be found at:
  https://www.smartmontools.org/

- An introduction of mcelog can be found at:
  https://mcelog.org/

- ipmitool how-to can be found at:
  http://linux.die.net/man/1/ipmitool

- An introduction to IPMI can be found at:
  http://www.intel.com/content/www/us/en/servers/ipmi/ipmi-home.html
- An introduction of Prometheus can be found at
  https://prometheus.io/docs/introduction/overview/

- An introduction of Grafana dashboard can be found at:
  https://grafana.com/grafana

- An introduction of Prometheus Alertmanager can be found at:
  https://prometheus.io/docs/alerting/alertmanager/

## Warnings, Cautions, and Notes

***Warning!*** *Warnings indicate conditions, which, if not observed, can cause personal injury.*

***Caution!*** *Cautions are included to help you avoid damaging your hardware or losing data.*

***Note!*** *Notes provide additional information.*

## We appreciate your input

Please let us know of any aspect of this product—including this manual—that could use improvement or correction. We appreciate your valuable input in helping make our products and documentation better.

Please send any related feedback to tse.ncg@advantech.com

# Table of Contents

## List of Figures

# List of Tables

## Glossary

| | |
|---|---|
| AHG | (Advanced) Health Guard |
| ahgc | (Advanced) Health Guard CLI Utility |
| AER | Advanced error reporting |
| API | Application programming interface |
| BMC | Baseboard management controller |
| CLI | Command-line interface |
| CPU | Central processing unit |
| HDD | Hard disk drive |
| HTTP | Hypertext Transfer Protocol |
| IPC | Inter-process communication |
| LCD | Liquid crystal display |
| MCE | Machine check exception |
| MIB | Management information base |
| NVMe | Non-Volatile memory express |
| PSU | Power supply unit |
| RAID | Redundant array of independent disks |
| REST | Representational state transfer |
| SMBIOS | System Management BIOS |
| SMTP | Simple Mail Transfer Protocol |
| S.M.A.R.T. | Self-monitoring analysis and reporting technology |
| SNMP | Simple Network Management Protocol |
| SSD | Solid state drive |
| URL | Uniform resource locator |

# 1. GETTING STARTED

## 1.1 Introduction

Server iManager as in Figure 1 is a Linux based software framework that integrates pre-existing SW utilities as well as new building blocks, focusing on health monitoring / diagnostics.

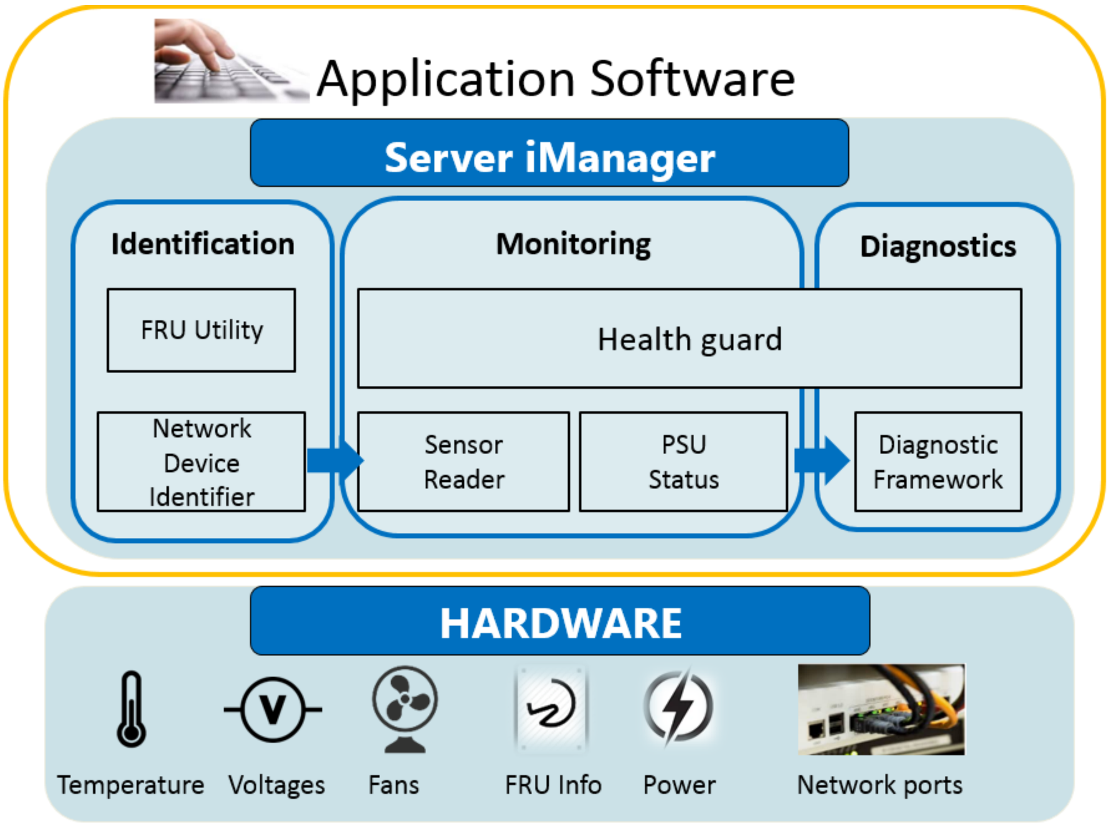


**Figure 1: Architecture of Server iManager**

Health Guard, the advanced component of Server iManager, is a hardware monitoring and notification solution for Advantech products. Similar to a security system, it runs as a daemon in x86 Linux and checks the output of several monitor utilities, issuing an alarm if an error is detected. The architecture of Health Guard is illustrated in Figure 2.

Health Guard includes these components:

1. A central daemon for controlling and communicating with all other components.
2. A configuration file parsing engine for setting Health Guard parameters.
3. A logging mechanism for logging messages to a file.
4. A collector for gathering hardware status information via different tools (e.g., smartctl, ipmitool).
5. An alarm engine for sending notifications via various methods.

6. RESTful API for CLI and RMM agent plugin which integrates with WISE-PaaS/EdgeSense. For details of WISE-PaaS/EdgeSense, please refer to the WISE-PaaS/EdgeSense user manual.
7. A standalone CLI for communicating with the daemon via RESTful APIs.
8. A summary utility for dumping system summaries.
9. Diagnostic framework is a software solution which provides a diagnostic capability for hardware.

Each component is described in the following diagram.

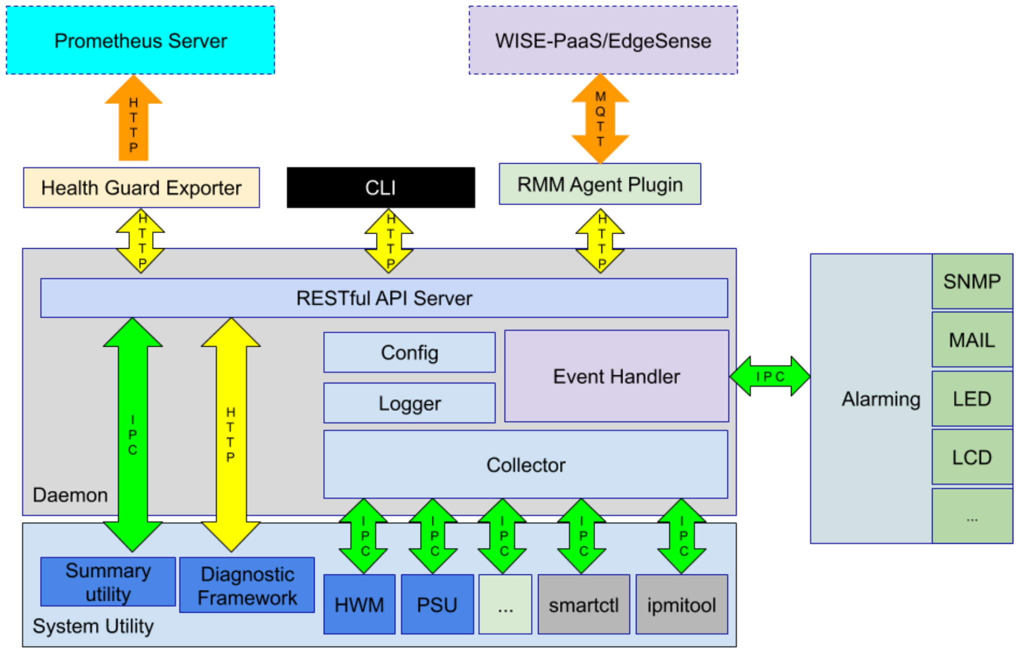


**Figure 2: Architecture of Health Guard**

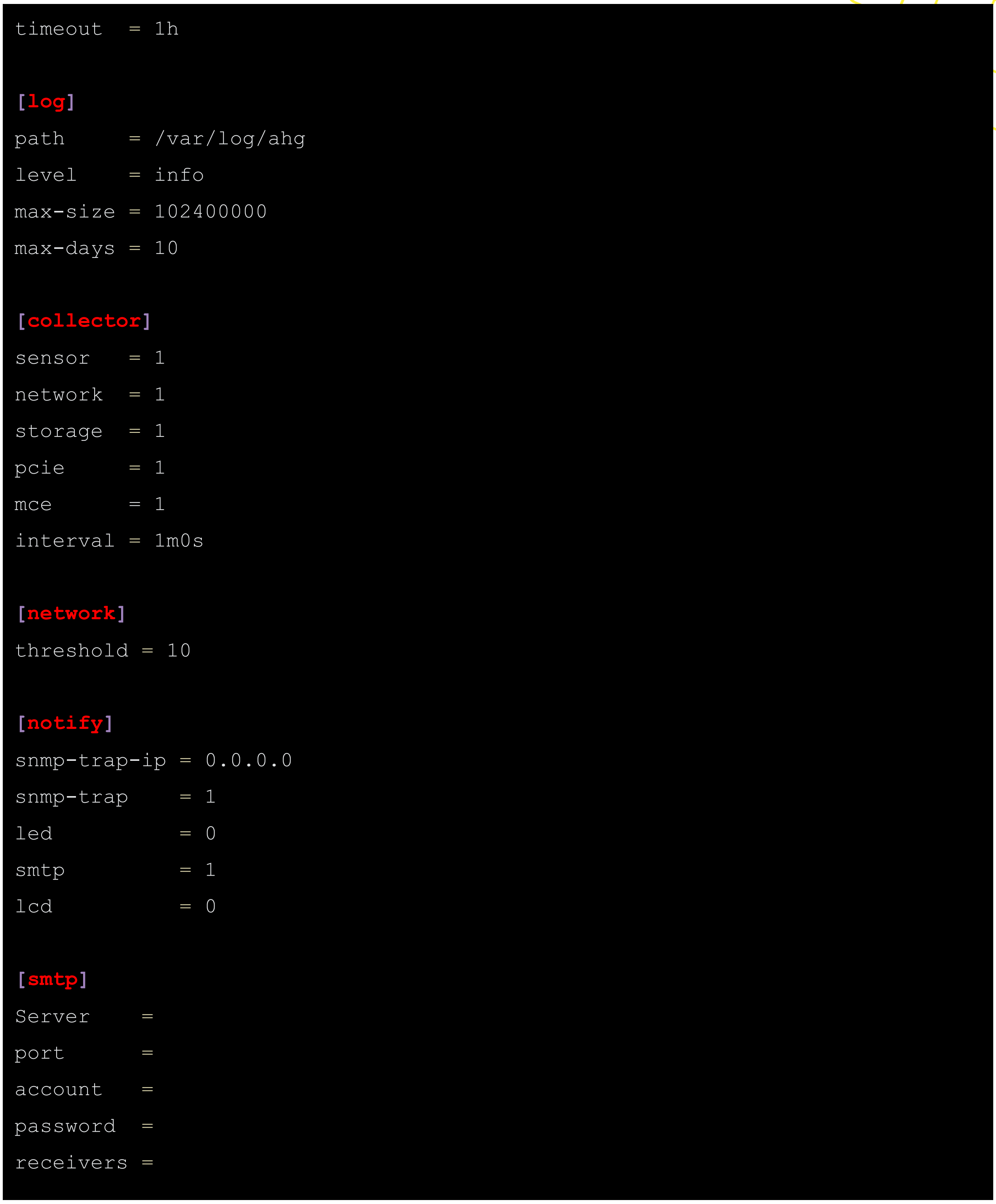
## 1.2 Installation

For the installation, please refer to “Server_iManager.pdf” in server_imanager.tgz.

## 1.3 Configuration

Health Guard contains a default configuration file called “config.ini,” which is located at the path “/opt/ahg.” The file contains some default settings that you can configure. The default “config.ini” is shown in Figure 3:

```
[rest]
user       = admin
port       = 8087
ip         = 0.0.0.0
```



```
timeout   = 1h

[log]
path      = /var/log/ahg
level     = info
max-size = 102400000
max-days = 10

[collector]
sensor    = 1
network   = 1
storage   = 1
pcie      = 1
mce       = 1
interval = 1m0s

[network]
threshold = 10

[notify]
snmp-trap-ip = 0.0.0.0
snmp-trap    = 1
led          = 0
smtp         = 1
lcd          = 0

[smtp]
Server     =
port       =
account    =
password   =
receivers  =
```

**Figure 3: Default Content in Config.ini**

The configuration file is in .ini format. The .ini format has section headers with the name of the section in square brackets. All keys after the section declaration are associated with that section. The default config.ini file contains sections, "rest," "log," "collector," "network," "notify," and "smtp," which is followed by the associated keys. For example, the "rest" section contains the keys “user,” “port,” “ip,” and “timeout.” The following subsections define the keys for each section. Users can configure the settings by using the command of ahgc. Please refer to Chapter 2 CLI for more detail usage of Health Guard CLI utility (ahgc).

### 1.3.1 Rest

| KEY | DESCRIPTION |
|---|---|
| User | Username used in REST authentication |
| Port | TCP port for the HTTP daemon (default: 8087) |
| IP | The IP address of the specific port that is allowed to access the server daemon via the REST API |
| Timeout | Authentication token expire time for the REST server |

**Table 1: Rest Section Key Definitions**

### 1.3.2 Log

| KEY | DESCRIPTION |
|---|---|
| Path | Log path (Default: /var/log/ahg) |
| Level | Log level; currently supported levels are “trace,” “info,” “warn,” “error,” and “fatal.” |
| Max-size | The maximize size of the log file (Default: 102,400,000; 100 MB) |
| Max-days | The keep time of the log file (default: 10; 10 days) |

**Table 2: Log Section Key Definitions**

### 1.3.3 Collector

| KEY | DESCRIPTION |
|---|---|
| Sensor | Flag that enables/disables the sensor collector (enable = 1 \| disable = 0) |
| Network | Flag that enables/disables the network collector (enable = 1 \| disable = 0) |
| Storage | Flag that enables/disables the storage collector (enable = 1 \| disable = 0) |
| PCIe | Flag of that enables/disables the PCIe collector (enable = 1 \| disable = 0) |
| MCE | Flag that enables/disables the MCE collector (enable = 1 \| disable = 0) |
| Interval | Refresh interval for sub-collectors (default: 1 min; written as “1m0s”). The range of interval is from 1 second to 1 hour. |

**Table** 3: **Collector Section Key Definitions**

### 1.3.4 Network

| KEY | DESCRIPTION |
|---|---|
| Threshold | Global threshold for error counter (see Chapter 3.2.2)<br>If the specified threshold is not set, this value will take effect. If the threshold is 0, checking for that counter will be skipped. |
| rx_errors | Threshold for Rx errors. If the threshold is 0, checking for that counter will be skipped. |
| tx_errors | Threshold for Tx errors. If the threshold is 0, checking for that counter will be skipped. |
| rx_dropped | Threshold for Rx dropped. If the threshold is 0, checking for that counter will be skipped. |
| tx_dropped | Threshold for Tx dropped. If the threshold is 0, checking for that counter will be skipped. |
| collisions | Threshold for Collisions. If the threshold is 0, checking for that counter will be skipped. |
| rx_length_errors | Threshold for Rx length errors. If the threshold is 0, checking for that counter will be skipped. |
| rx_over_errors | Threshold for Rx over errors. If the threshold is 0, checking for that counter will be skipped. |
| rx_crc_errors | Threshold for Rx CRC errors. If the threshold is 0, checking for that counter will be skipped. |

|  |  |
|---|---|
| rx_frame_errors | Threshold for Rx frame errors. If the threshold is 0, checking for that counter will be skipped. |
| rx_fifo_errors | Threshold for Rx fifo errors. If the threshold is 0, checking for that counter will be skipped. |
| rx_missed_errors | Threshold for Rx missed errors. If the threshold is 0, checking for that counter will be skipped. |
| tx_aborted_errors | Threshold for Tx aborted errors. If the threshold is 0, checking for that counter will be skipped. |
| tx_carrier_errors | Threshold for Tx carrier errors. If the threshold is 0, checking for that counter will be skipped. |
| tx_fifo_errors | Threshold for Tx FIFO errors. If the threshold is 0, checking for that counter will be skipped. |
| tx_heartbeat_errors | Threshold for Tx heartbeat errors. If the threshold is 0, checking for that counter will be skipped. |
| tx_window_errors | Threshold for Tx window errors. If the threshold is 0, checking for that counter will be skipped. |

**Table 4: Network Section Key Definitions**

### 1.3.5 Notify

| KEY | DESCRIPTION |
|---|---|
| SNMP-trap-ip | IP address of the SNMP trap receiver |
| SNMP-trap | Flag that enables/disables the SNMP-trap “notify” method (enable = 1 \| disable = 0). If SNMP-trap = 1, the monitor will send an SNMP-trap when any events are generated. |
| LED | Flag that enables/disables the LED “notify” method (enable = 1 \| disable = 0). If LED = 1, Health Guard will turn on the LED whenever a platform error occurs. |
| SMTP | Flag that enables/disables the SMTP “notify” method (enable = 1 \| disable = 0). If SMTP = 1, then Health Guard will send an SMTP email when any events are generated. |
| LCD | Flag that enables/disables the LCD “notify” method (enable = 1 \| disable = 0). If LCD = 1, then Health Guard will display an error icon on the LCD when an error occurs on the platform. |

**Table 5: Notify Section Key Definitions**

### 1.3.6 SMTP

The SMTP section contains descriptions of these keys: account, password, server, port, and receivers. This section must be configured when SMTP = 1 is in the notify section.

| KEY | DESCRIPTION |
|---|---|
| Account | Sender mail account |
| Password | Password of mail account |
| Server | SMTP server of sender account; this can be a domain name or IP address |
| Port | SMTP server port |
| Receivers | Recipients accounts; this can be multiple accounts (e.g., receivers = “receiver1@domain1, receiver2@domain2”) |

**Table 6: SMTP Section Key Definitions**

# 2. CLI

## 2.1 Introduction

The CLI utility "ahgc" interacts with the RESTful API to query information or to configure Health Guard.

## 2.2 Usage

The syntax of the ahgc command is as follows:

***# ahgc [global options] command [command options] [arguments...]***

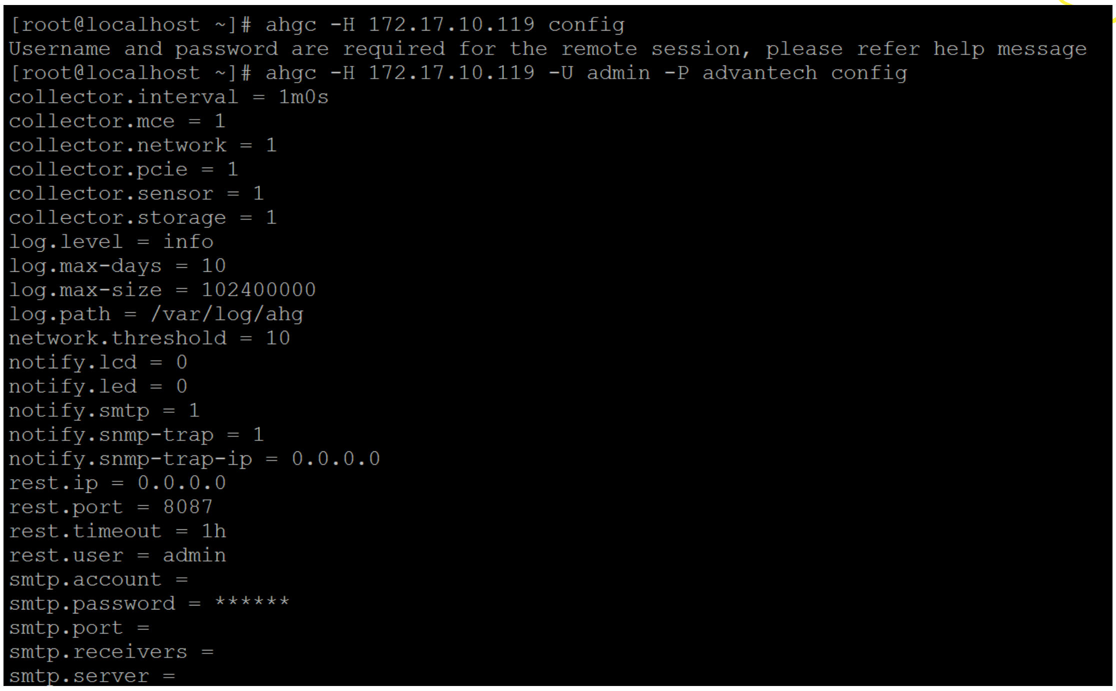
For details on the usage of each command, please refer to the following subsections.

### 2.2.1 Global Options Commands

| Global Options | | |
|---|---|---|
| **Option** | **Option in short** | **Description** |
| --help | -h | Shows help |
| --version | -v | Prints the version |
| --hostname hostname | -H hostname | Remote hostname for session *(default 127.0.0.1)* |
| --port port, | -p port | Remote port for session *(default 8087)* |
| --username username | -U username | Remote session username |
| --password password | -P password | Remote session password, default password is "advantech" |

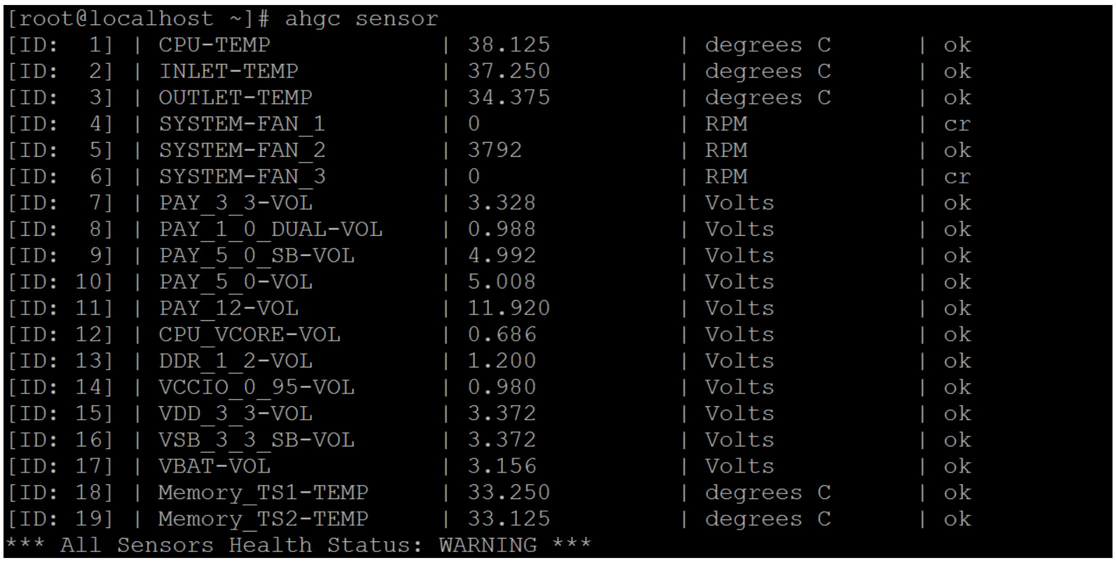
**Table 7: Global options**

The username and password must be specified to access Health Guard remotely. The default hostname is "127.0.0.1" and the default port is "8087"; if you modify the port settings in the config.ini, you will need to specify the port when executing the ahgc command no matter via remote or local sessions. Otherwise, execution from a remote session will fail. The examples of remote and local sessions are as below:



```
[root@localhost ~]# ahgc -H 172.17.10.119 config
Username and password are required for the remote session, please refer help message
[root@localhost ~]# ahgc -H 172.17.10.119 -U admin -P advantech config
collector.interval = 1m0s
collector.mce = 1
collector.network = 1
collector.pcie = 1
collector.sensor = 1
collector.storage = 1
log.level = info
log.max-days = 10
log.max-size = 102400000
log.path = /var/log/ahg
network.threshold = 10
notify.lcd = 0
notify.led = 0
notify.smtp = 1
notify.snmp-trap = 1
notify.snmp-trap-ip = 0.0.0.0
rest.ip = 0.0.0.0
rest.port = 8087
rest.timeout = 1h
rest.user = admin
smtp.account =
smtp.password = ******
smtp.port =
smtp.receivers =
smtp.server =
```

**Figure 4: Example of a Remote Session**



```
[root@localhost ~]# ahgc sensor
[ID:  1] | CPU-TEMP             | 38.125          | degrees C      | ok
[ID:  2] | INLET-TEMP           | 37.250          | degrees C      | ok
[ID:  3] | OUTLET-TEMP          | 34.375          | degrees C      | ok
[ID:  4] | SYSTEM-FAN_1         | 0               | RPM            | cr
[ID:  5] | SYSTEM-FAN_2         | 3792            | RPM            | ok
[ID:  6] | SYSTEM-FAN_3         | 0               | RPM            | cr
[ID:  7] | PAY_3_3-VOL          | 3.328           | Volts          | ok
[ID:  8] | PAY_1_0_DUAL-VOL     | 0.988           | Volts          | ok
[ID:  9] | PAY_5_0_SB-VOL       | 4.992           | Volts          | ok
[ID: 10] | PAY_5_0-VOL          | 5.008           | Volts          | ok
[ID: 11] | PAY_12-VOL           | 11.920          | Volts          | ok
[ID: 12] | CPU_VCORE-VOL        | 0.686           | Volts          | ok
[ID: 13] | DDR_1_2-VOL          | 1.200           | Volts          | ok
[ID: 14] | VCCIO_0_95-VOL       | 0.980           | Volts          | ok
[ID: 15] | VDD_3_3-VOL          | 3.372           | Volts          | ok
[ID: 16] | VSB_3_3_SB-VOL       | 3.372           | Volts          | ok
[ID: 17] | VBAT-VOL             | 3.156           | Volts          | ok
[ID: 18] | Memory_TS1-TEMP      | 33.250          | degrees C      | ok
[ID: 19] | Memory_TS2-TEMP      | 33.125          | degrees C      | ok
*** All Sensors Health Status: WARNING ***
```

**Figure 5: Example of a Local Session**

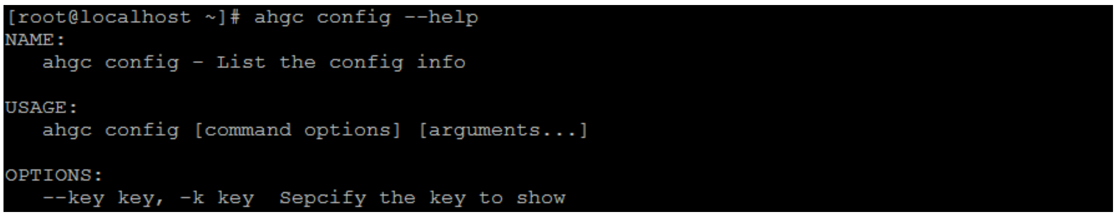
### 2.2.2 Command Options

| **Command Name** | **Description** |
|---|---|
| config | Lists the config information |
| sensor | Lists the sensor information |
| storage | Lists the storage information |

| | |
|---|---|
| network | Lists the network information |
| mce | Lists the MCE error messages |
| pcie | Lists the PCIe information |
| event | Lists the event history |
| set | Sets the value of the key in daemon configuration |
| status | Lists the status of collectors |
| summary | Gathers summary information |
| test | Verifies the configuration file |
| restart | Restarts Server iManager daemon |

**Table 8: Commands**

You can add the command option "--help, -h" to obtain details on the usage of each command, as shown in Figure 6:



```
[root@localhost ~]# ahgc config --help
NAME:
   ahgc config - List the config info

USAGE:
   ahgc config [command options] [arguments...]

OPTIONS:
   --key key, -k key  Sepcify the key to show
```

**Figure 6: Usage of "config" Command**

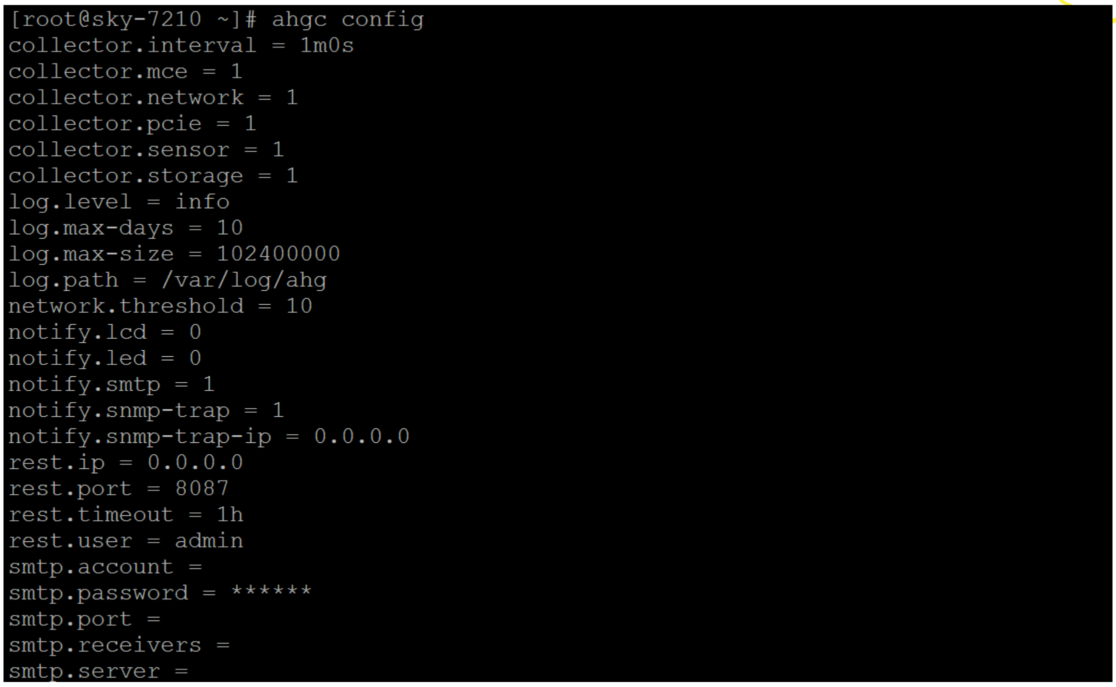
### 2.2.2.1 Config Command

| **Command: config** | | |
|---|---|---|
| **Options** | **Options in Short Form** | **Description** |
| --key key | -k key | Specifies the <section>.<key> in config.ini to show |

**Table 9: Usage of "config" Command**

The "config" command lists the configurations, as shown in Figure 7:

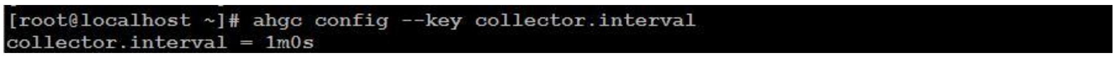
***Note!*** *For security purposes, the password that CLI prints will always be "******".*



```
[root@sky-7210 ~]# ahgc config
collector.interval = 1m0s
collector.mce = 1
collector.network = 1
collector.pcie = 1
collector.sensor = 1
collector.storage = 1
log.level = info
log.max-days = 10
log.max-size = 102400000
log.path = /var/log/ahg
network.threshold = 10
notify.lcd = 0
notify.led = 0
notify.smtp = 1
notify.snmp-trap = 1
notify.snmp-trap-ip = 0.0.0.0
rest.ip = 0.0.0.0
rest.port = 8087
rest.timeout = 1h
rest.user = admin
smtp.account =
smtp.password = ******
smtp.port =
smtp.receivers =
smtp.server =
```

**Figure 7: "config" Command**

You can get the specific configuration by adding the parameter, "--key <section>.<name> "as shown in Figure 8 or "-k <section>.<name>". For details on config.ini, please refer to Chapter 1.3.



```
[root@localhost ~]# ahgc config --key collector.interval
collector.interval = 1m0s
```

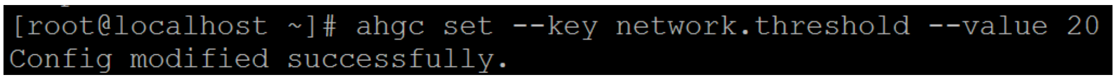
**Figure 8: Config Command with Key Options**

### 2.2.2.2 Set Command

| Command: set | | |
|---|---|---|
| **Options** | **Options in Short Form** | **Description** |
| --key key | -k key | Specifies the section.key in config.ini to set |
| --value value | -v value | Specifies the value of section.key. |

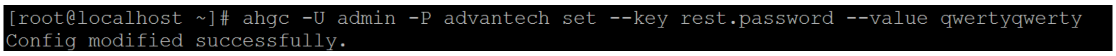
**Table 10: Usage of the Set Command**

You can set the specific configuration by adding the "--key, -k" & "--value, -v" options, as shown in Figure 9 and Figure 10.



```
[root@localhost ~]# ahgc set --key network.threshold --value 20
Config modified successfully.
```

**Figure 9: "config" Command with Key and Value Options**



```
[root@localhost ~]# ahgc -U admin -P advantech set --key rest.password --value qwertyqwerty
Config modified successfully.
```

**Figure 10: "config" Command with Key and Value Options**

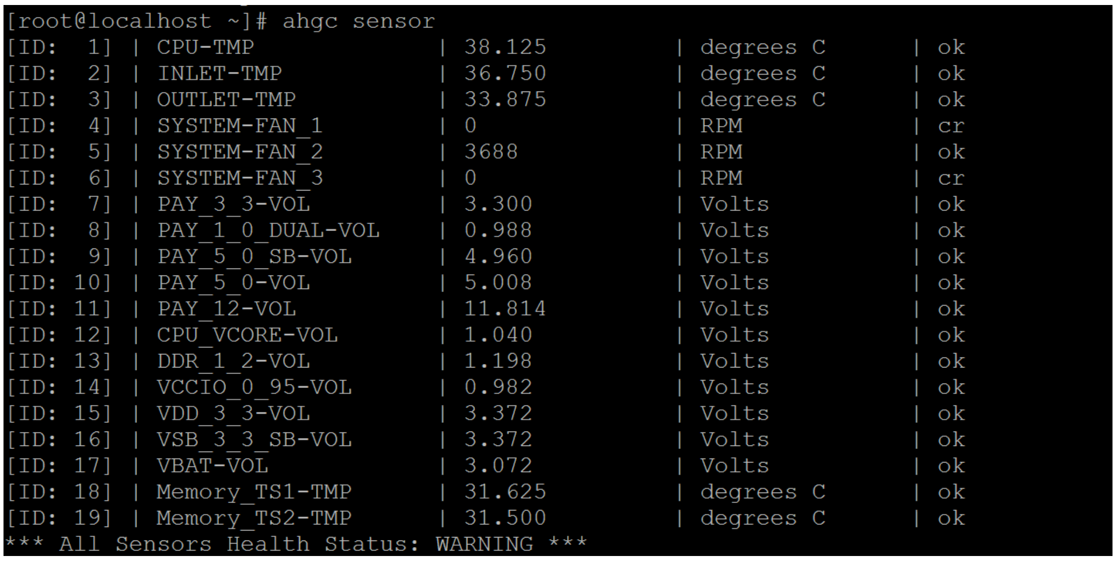
***Note!*** *For security purposes, please specify the username and password when setting the password for both local and remote sessions. The minimal length of password is 8 and the maximum length is 20.*

### 2.2.2.3 Sensor Command

| Command: sensor | | |
|---|---|---|
| **Options** | **Options in Short Form** | **Description** |
| --name name | -n name | Specifies the sensor to show, for example, users can get the information of sensor CPU-TEMP from the command "ahgc sensor --name CPU-TEMP". |

**Table 11: Usage of the Sensor Command**

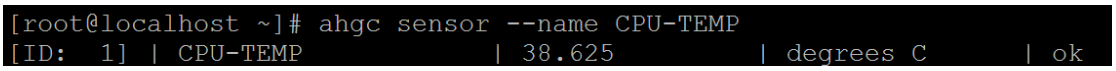
The sensor command lists the sensor information, as shown in Figure 11:



```
[root@localhost ~]# ahgc sensor
[ID:  1] | CPU-TMP              | 38.125        | degrees C       | ok
[ID:  2] | INLET-TMP            | 36.750        | degrees C       | ok
[ID:  3] | OUTLET-TMP           | 33.875        | degrees C       | ok
[ID:  4] | SYSTEM-FAN_1         | 0             | RPM             | cr
[ID:  5] | SYSTEM-FAN_2         | 3688          | RPM             | ok
[ID:  6] | SYSTEM-FAN_3         | 0             | RPM             | cr
[ID:  7] | PAY_3_3-VOL          | 3.300         | Volts           | ok
[ID:  8] | PAY_1_0_DUAL-VOL     | 0.988         | Volts           | ok
[ID:  9] | PAY_5_0_SB-VOL       | 4.960         | Volts           | ok
[ID: 10] | PAY_5_0-VOL          | 5.008         | Volts           | ok
[ID: 11] | PAY_12-VOL           | 11.814        | Volts           | ok
[ID: 12] | CPU_VCORE-VOL        | 1.040         | Volts           | ok
[ID: 13] | DDR_1_2-VOL          | 1.198         | Volts           | ok
[ID: 14] | VCCIO_0_95-VOL       | 0.982         | Volts           | ok
[ID: 15] | VDD_3_3-VOL          | 3.372         | Volts           | ok
[ID: 16] | VSB_3_3_SB-VOL       | 3.372         | Volts           | ok
[ID: 17] | VBAT-VOL             | 3.072         | Volts           | ok
[ID: 18] | Memory_TS1-TMP       | 31.625        | degrees C       | ok
[ID: 19] | Memory_TS2-TMP       | 31.500        | degrees C       | ok
*** All Sensors Health Status: WARNING ***
```

**Figure 11: Sensor Command**

You can get the specific sensor data by adding the "--name, -n" option with the sensor name, as shown in Figure 12. For details on the sensor command, please refer to Chapter 3.2.1.



```
[root@localhost ~]# ahgc sensor --name CPU-TEMP
[ID:  1] | CPU-TEMP             | 38.625        | degrees C       | ok
```

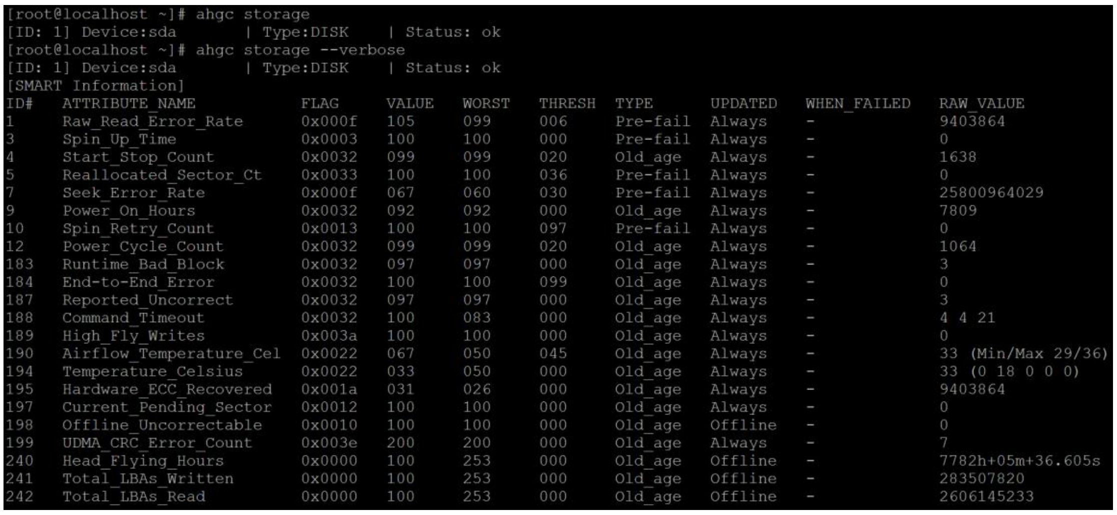
**Figure 12: Sensor Command with Name Option**

### 2.2.2.4 Storage Command

| Command: storage | | |
|---|---|---|
| **Options** | **Options in Short Form** | **Description** |
| --name name | -n name | Specifies the storage to show, for example, users can get the information of storage sda from command “ahgc storage --name sda”. |
| --verbose | -v | Verbose output |

**Table 12: Usage of the Storage Command**

The storage command lists the storage information. You can get specific storage data by adding the “--name, -n” option with the storage name. You can also get the verbose output by adding the “--verbose, -v” option. An example of the storage command is shown in Figure 13. For details on the storage command, please refer to Chapter 3.2.3.



```
[root@localhost ~]# ahgc storage
[ID: 1] Device:sda       | Type:DISK    | Status: ok
[root@localhost ~]# ahgc storage --verbose
[ID: 1] Device:sda       | Type:DISK    | Status: ok
[SMART Information]
ID#    ATTRIBUTE_NAME            FLAG     VALUE   WORST   THRESH  TYPE      UPDATED   WHEN_FAILED   RAW_VALUE
1      Raw_Read_Error_Rate       0x000f   105     099     006     Pre-fail  Always    -             9403864
3      Spin_Up_Time              0x0003   100     100     000     Pre-fail  Always    -             0
4      Start_Stop_Count          0x0032   099     099     020     Old_age   Always    -             1638
5      Reallocated_Sector_Ct     0x0033   100     100     036     Pre-fail  Always    -             0
7      Seek_Error_Rate           0x000f   067     060     030     Pre-fail  Always    -             25800964029
9      Power_On_Hours            0x0032   092     092     000     Old_age   Always    -             7809
10     Spin_Retry_Count          0x0013   100     100     097     Pre-fail  Always    -             0
12     Power_Cycle_Count         0x0032   099     099     020     Old_age   Always    -             1064
183    Runtime_Bad_Block         0x0032   097     097     000     Old_age   Always    -             3
184    End-to-End_Error          0x0032   100     100     099     Old_age   Always    -             0
187    Reported_Uncorrect        0x0032   097     097     000     Old_age   Always    -             3
188    Command_Timeout           0x0032   100     083     000     Old_age   Always    -             4 4 21
189    High_Fly_Writes           0x003a   100     100     000     Old_age   Always    -             0
190    Airflow_Temperature_Cel   0x0022   067     050     045     Old_age   Always    -             33 (Min/Max 29/36)
194    Temperature_Celsius       0x0022   033     050     000     Old_age   Always    -             33 (0 18 0 0 0)
195    Hardware_ECC_Recovered    0x001a   031     026     000     Old_age   Always    -             9403864
197    Current_Pending_Sector    0x0012   100     100     000     Old_age   Always    -             0
198    Offline_Uncorrectable     0x0010   100     100     000     Old_age   Offline   -             0
199    UDMA_CRC_Error_Count      0x003e   200     200     000     Old_age   Always    -             7
240    Head_Flying_Hours         0x0000   100     253     000     Old_age   Offline   -             7782h+05m+36.605s
241    Total_LBAs_Written        0x0000   100     253     000     Old_age   Offline   -             283507820
242    Total_LBAs_Read           0x0000   100     253     000     Old_age   Offline   -             2606145233
```

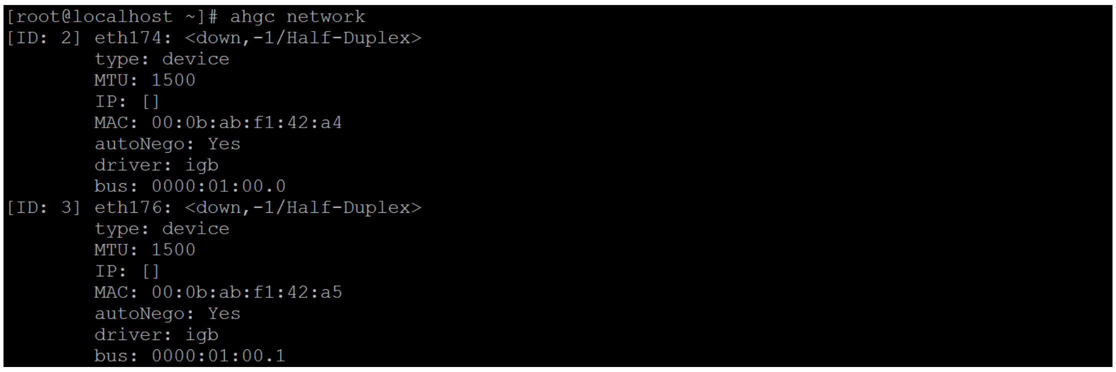
**Figure 13: Storage Command**

### 2.2.2.5 Network Command

| Command: network | | |
|---|---|---|
| **Options** | **Options in Short Form** | **Description** |
| --nic nic | -n nic | Specifies the nic to show, for example, user can get the information of nic eth2 from command "ahgc network --nic eth2". |
| --verbose | -v | Verbose output |

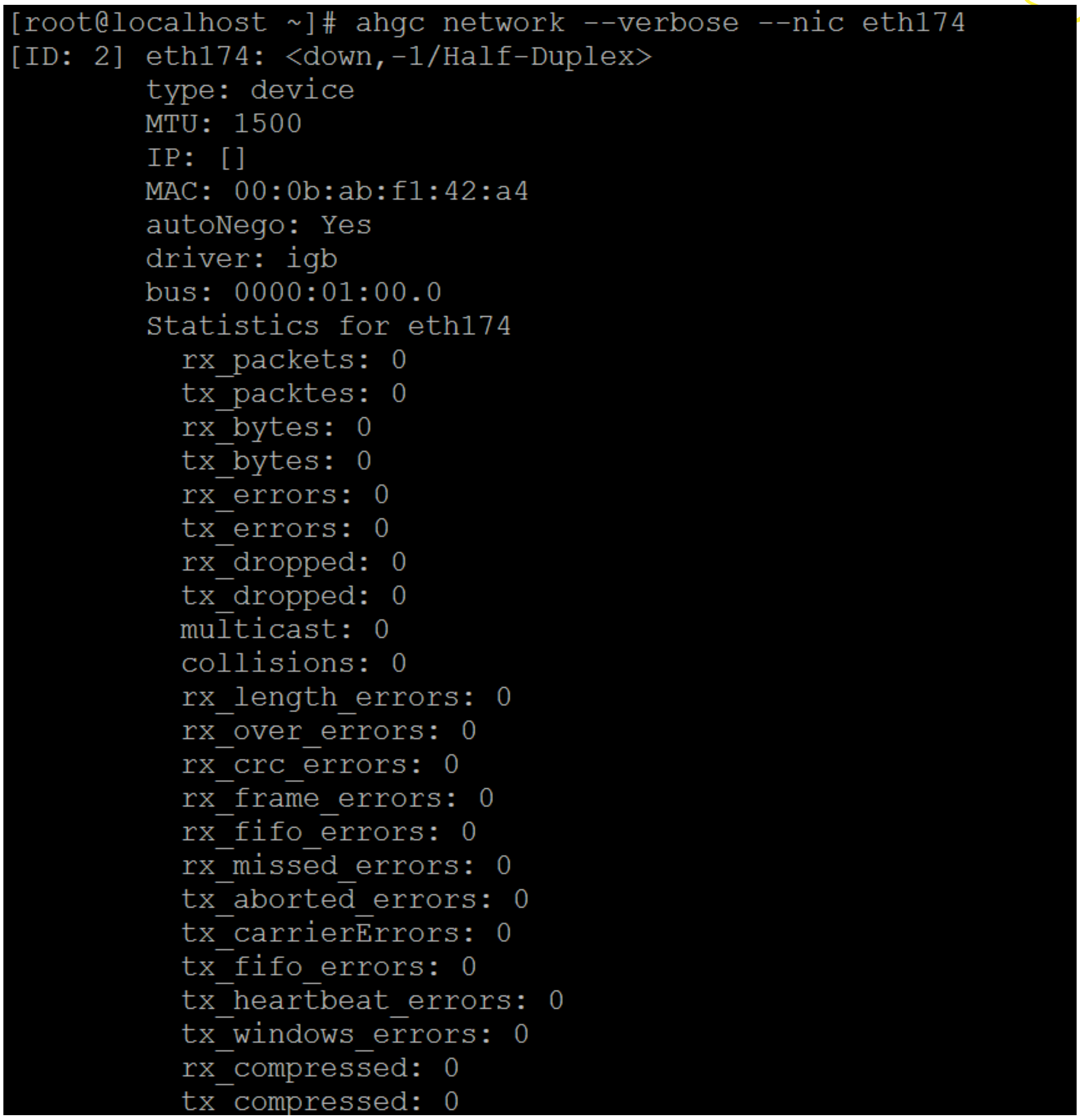
**Table 13: Usage of the Network Command**

The network command lists the network information. You can get specific network data by adding the "--nic, -n" option with the network name. You can also get the verbose output by adding the "--verbose, -v" option. Examples of the network command are shown in Figure 14 and Figure 15. For details on the network command, please refer to Chapter 3.2.2.



```
[root@localhost ~]# ahgc network
[ID: 2] eth174: <down,-1/Half-Duplex>
        type: device
        MTU: 1500
        IP: []
        MAC: 00:0b:ab:f1:42:a4
        autoNego: Yes
        driver: igb
        bus: 0000:01:00.0
[ID: 3] eth176: <down,-1/Half-Duplex>
        type: device
        MTU: 1500
        IP: []
        MAC: 00:0b:ab:f1:42:a5
        autoNego: Yes
        driver: igb
        bus: 0000:01:00.1
```

**Figure 14: Network Command**



```
[root@localhost ~]# ahgc network --verbose --nic eth174
[ID: 2] eth174: <down,-1/Half-Duplex>
        type: device
        MTU: 1500
        IP: []
        MAC: 00:0b:ab:f1:42:a4
        autoNego: Yes
        driver: igb
        bus: 0000:01:00.0
        Statistics for eth174
          rx_packets: 0
          tx_packtes: 0
          rx_bytes: 0
          tx_bytes: 0
          rx_errors: 0
          tx_errors: 0
          rx_dropped: 0
          tx_dropped: 0
          multicast: 0
          collisions: 0
          rx_length_errors: 0
          rx_over_errors: 0
          rx_crc_errors: 0
          rx_frame_errors: 0
          rx_fifo_errors: 0
          rx_missed_errors: 0
          tx_aborted_errors: 0
          tx_carrierErrors: 0
          tx_fifo_errors: 0
          tx_heartbeat_errors: 0
          tx_windows_errors: 0
          rx_compressed: 0
          tx_compressed: 0
```

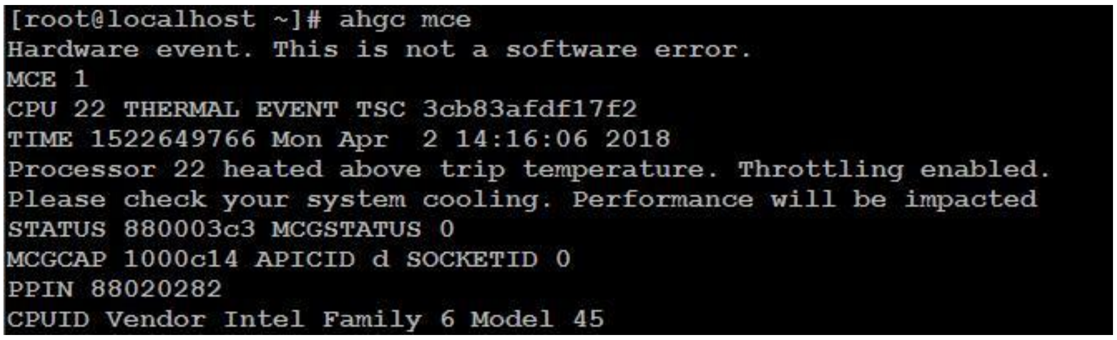
**Figure 15: Network Command with Name Option**

### 2.2.2.6 Mce Command

| **Command: mce** | | |
|---|---|---|
| **Options** | **Options in Short Form** | **Description** |
| N/A | N/A | N/A |

**Table 14: Usage of the MCE Command**

The mce command lists any MCE errors. An example of the mce command is shown in Figure 16. For details on the mce command, please refer to Chapter 3.2.5.



```
[root@localhost ~]# ahgc mce
Hardware event. This is not a software error.
MCE 1
CPU 22 THERMAL EVENT TSC 3cb83afdf17f2
TIME 1522649766 Mon Apr  2 14:16:06 2018
Processor 22 heated above trip temperature. Throttling enabled.
Please check your system cooling. Performance will be impacted
STATUS 880003c3 MCGSTATUS 0
MCGCAP 1000c14 APICID d SOCKETID 0
PPIN 88020282
CPUID Vendor Intel Family 6 Model 45
```

**Figure 16: MCE Command**

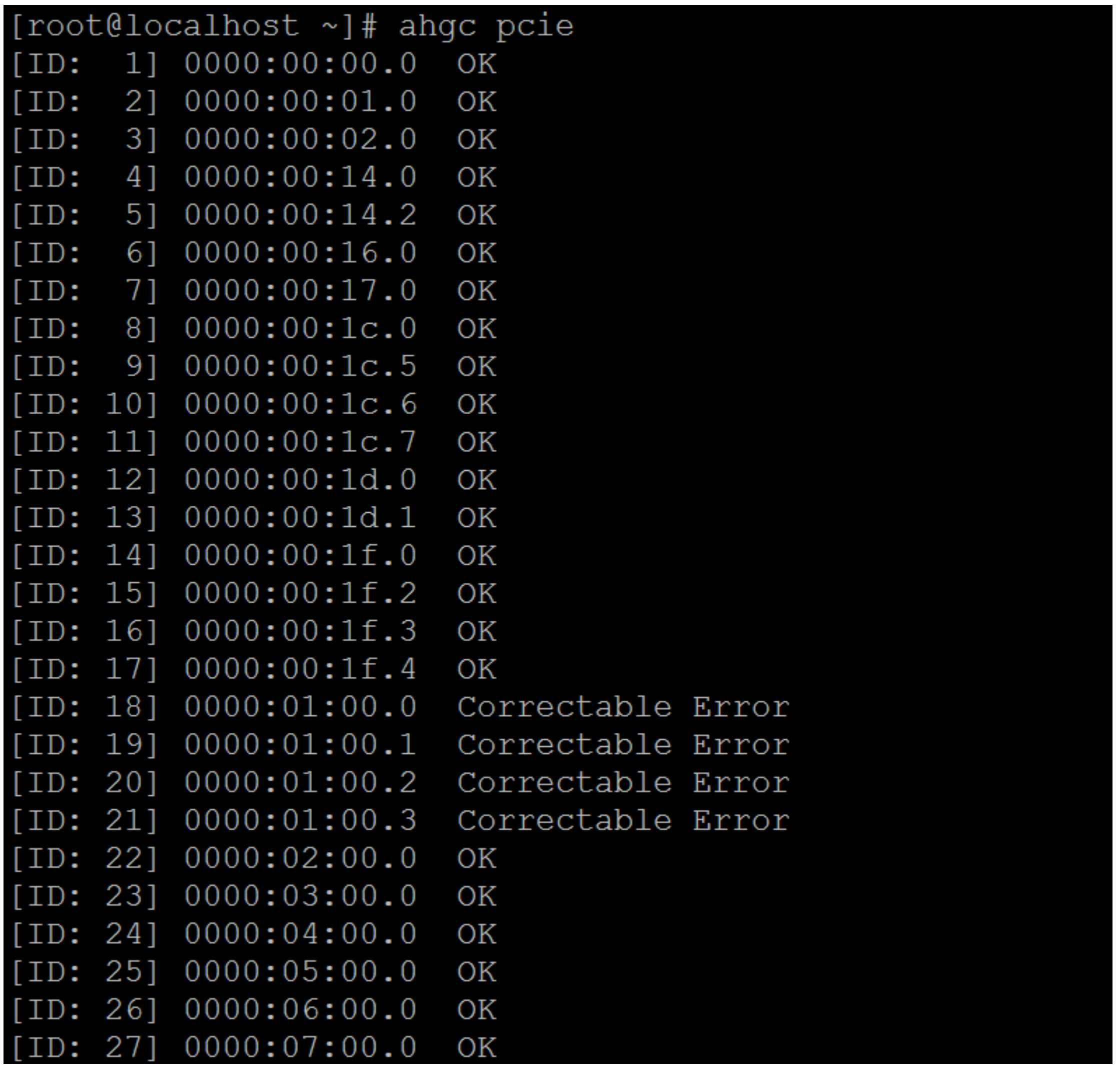
***Note!*** *The mce command is based on the utility “mcelog”; the system needs to run mcelog as a daemon.*

### 2.2.2.7 PCIe Command

| Command: pcie | | |
|---|---|---|
| **Options** | **Options in Short Form** | **Description** |
| --bus bus | -b bus | Specify the bus to show, for example, user can get the information of single PCIe device from command "ahgc pcie --bus 0000:00:1f.3". |

**Table 15: Usage of the PCIe Command**

The pcie command lists the PCIe information. You can get the specific PCIe data by adding the "--bus, -b" option with the bus information. An example of the pcie command is shown in Figure 17. For details of the pcie command, please refer to Chapter 3.2.4.



```
[root@localhost ~]# ahgc pcie
[ID:  1] 0000:00:00.0  OK
[ID:  2] 0000:00:01.0  OK
[ID:  3] 0000:00:02.0  OK
[ID:  4] 0000:00:14.0  OK
[ID:  5] 0000:00:14.2  OK
[ID:  6] 0000:00:16.0  OK
[ID:  7] 0000:00:17.0  OK
[ID:  8] 0000:00:1c.0  OK
[ID:  9] 0000:00:1c.5  OK
[ID: 10] 0000:00:1c.6  OK
[ID: 11] 0000:00:1c.7  OK
[ID: 12] 0000:00:1d.0  OK
[ID: 13] 0000:00:1d.1  OK
[ID: 14] 0000:00:1f.0  OK
[ID: 15] 0000:00:1f.2  OK
[ID: 16] 0000:00:1f.3  OK
[ID: 17] 0000:00:1f.4  OK
[ID: 18] 0000:01:00.0  Correctable Error
[ID: 19] 0000:01:00.1  Correctable Error
[ID: 20] 0000:01:00.2  Correctable Error
[ID: 21] 0000:01:00.3  Correctable Error
[ID: 22] 0000:02:00.0  OK
[ID: 23] 0000:03:00.0  OK
[ID: 24] 0000:04:00.0  OK
[ID: 25] 0000:05:00.0  OK
[ID: 26] 0000:06:00.0  OK
[ID: 27] 0000:07:00.0  OK
```

**Figure 17: PCIe Command**

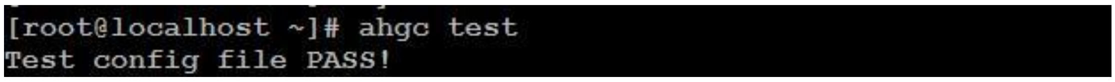
#### 2.2.2.8 Test Command

| **Command: test** | | |
|---|---|---|
| **Options** | **Options in Short Form** | **Description** |
| N/A | N/A | N/A |

**Table 16: Usage of the Test Command**

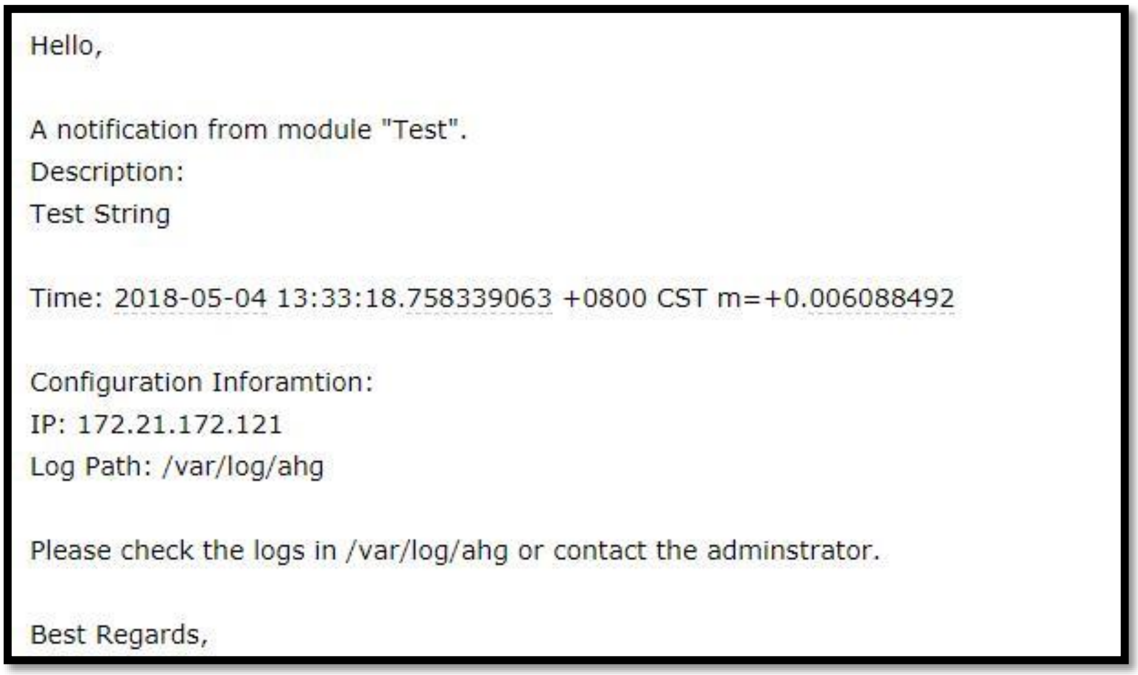
The test command tests the notification and SMTP settings in config.ini. An example of the test command is shown in Figure 18. For details of the test command, please refer to Chapter 4.

The test command gives a message indicating whether the testing result is a pass or fail. If the test result is a pass, you will receive an email, as shown in Figure 19.



```
[root@localhost ~]# ahgc test
Test config file PASS!
```

**Figure 18: Test Command**



Hello,

A notification from module "Test".
Description:
Test String

Time: 2018-05-04 13:33:18.758339063 +0800 CST m=+0.006088492

Configuration Inforamtion:
IP: 172.21.172.121
Log Path: /var/log/ahg

Please check the logs in /var/log/ahg or contact the adminstrator.

Best Regards,

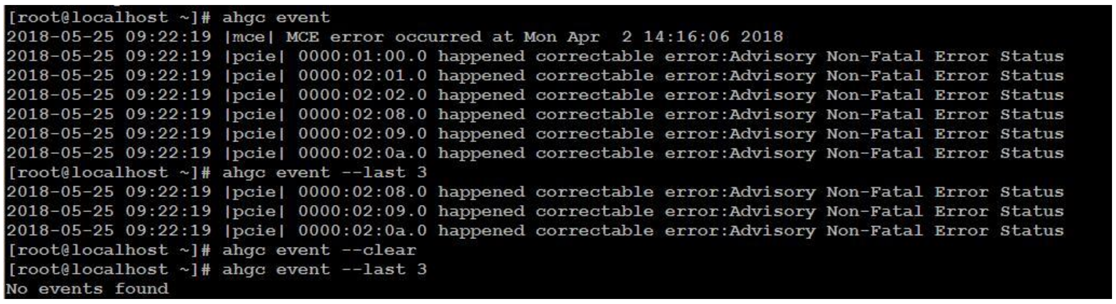
**Figure 19: Email from the Test Command**

### 2.2.2.9 Event Command

| Command: event | | |
|---|---|---|
| **Options** | **Options in short** | **Description** |
| --last number | -l number | Specifies the number of last event history entries |
| --clear | -c | Clears the event history |

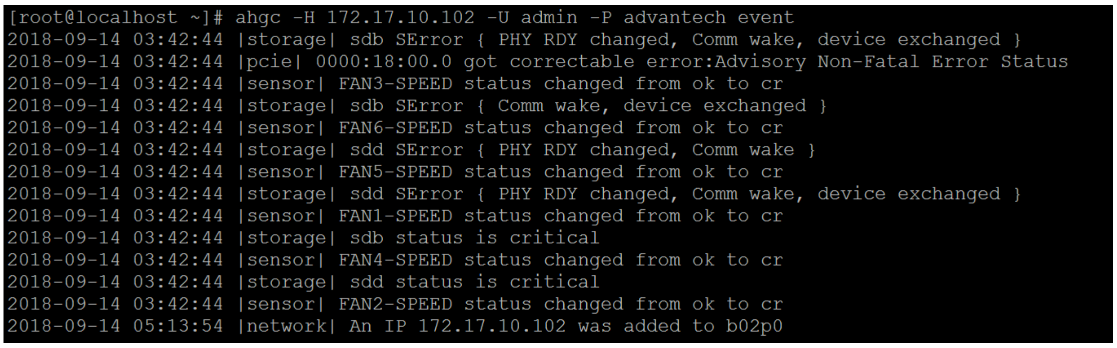
**Table 17: Usage of the Event Command**

The event command lists the last event history entries. You can specify the number of last entries by adding the "--last, -l" option (default: 100; range: 1~1000). You can also clear the event history by adding the "--clear, -c" option. An example of the event command is shown in Figure 20 (local) and Figure 21 (remote). For details on the event command, please refer to Chapter 3.3.



```
[root@localhost ~]# ahgc event
2018-05-25 09:22:19 |mce| MCE error occurred at Mon Apr  2 14:16:06 2018
2018-05-25 09:22:19 |pcie| 0000:01:00.0 happened correctable error:Advisory Non-Fatal Error Status
2018-05-25 09:22:19 |pcie| 0000:02:01.0 happened correctable error:Advisory Non-Fatal Error Status
2018-05-25 09:22:19 |pcie| 0000:02:02.0 happened correctable error:Advisory Non-Fatal Error Status
2018-05-25 09:22:19 |pcie| 0000:02:08.0 happened correctable error:Advisory Non-Fatal Error Status
2018-05-25 09:22:19 |pcie| 0000:02:09.0 happened correctable error:Advisory Non-Fatal Error Status
2018-05-25 09:22:19 |pcie| 0000:02:0a.0 happened correctable error:Advisory Non-Fatal Error Status
[root@localhost ~]# ahgc event --last 3
2018-05-25 09:22:19 |pcie| 0000:02:08.0 happened correctable error:Advisory Non-Fatal Error Status
2018-05-25 09:22:19 |pcie| 0000:02:09.0 happened correctable error:Advisory Non-Fatal Error Status
2018-05-25 09:22:19 |pcie| 0000:02:0a.0 happened correctable error:Advisory Non-Fatal Error Status
[root@localhost ~]# ahgc event --clear
[root@localhost ~]# ahgc event --last 3
No events found
```

**Figure 20: Event Command from a Local Session**



```
[root@localhost ~]# ahgc -H 172.17.10.102 -U admin -P advantech event
2018-09-14 03:42:44 |storage| sdb SError { PHY RDY changed, Comm wake, device exchanged }
2018-09-14 03:42:44 |pcie| 0000:18:00.0 got correctable error:Advisory Non-Fatal Error Status
2018-09-14 03:42:44 |sensor| FAN3-SPEED status changed from ok to cr
2018-09-14 03:42:44 |storage| sdb SError { Comm wake, device exchanged }
2018-09-14 03:42:44 |sensor| FAN6-SPEED status changed from ok to cr
2018-09-14 03:42:44 |storage| sdd SError { PHY RDY changed, Comm wake }
2018-09-14 03:42:44 |sensor| FAN5-SPEED status changed from ok to cr
2018-09-14 03:42:44 |storage| sdd SError { PHY RDY changed, Comm wake, device exchanged }
2018-09-14 03:42:44 |sensor| FAN1-SPEED status changed from ok to cr
2018-09-14 03:42:44 |storage| sdb status is critical
2018-09-14 03:42:44 |sensor| FAN4-SPEED status changed from ok to cr
2018-09-14 03:42:44 |storage| sdd status is critical
2018-09-14 03:42:44 |sensor| FAN2-SPEED status changed from ok to cr
2018-09-14 05:13:54 |network| An IP 172.17.10.102 was added to b02p0
```

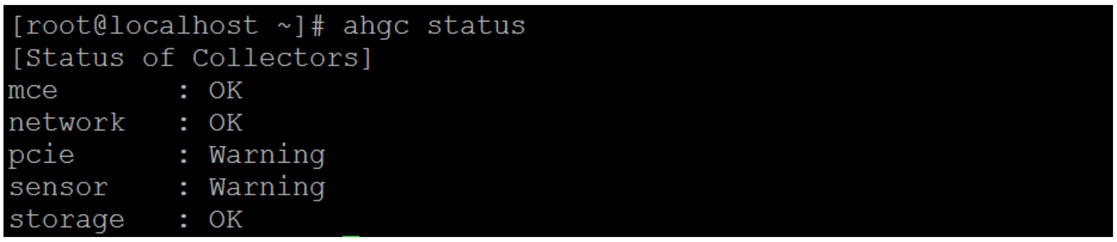
**Figure 21: Event Command from a Remote Session**

#### 2.2.2.10 Status Command

| Command: status | | |
|---|---|---|
| **Options** | **Options in short** | **Description** |
| N/A | N/A | Lists status of collectors |

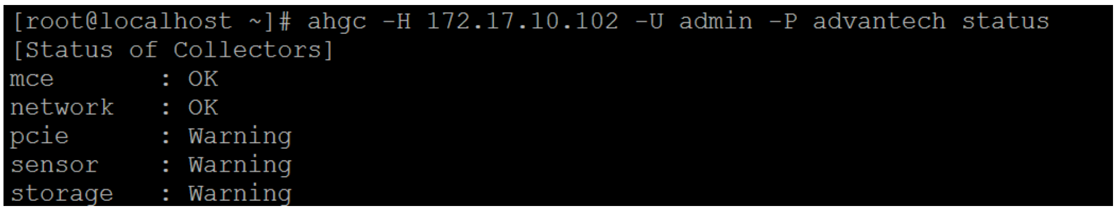
**Table 18: Usage of the Event Command**

The status command lists the status of collectors. An example of the status command is shown in Figure 22 (local) and Figure 23 (remote).



```
[root@localhost ~]# ahgc status
[Status of Collectors]
mce         : OK
network     : OK
pcie        : Warning
sensor      : Warning
storage     : OK
```

**Figure 22: Status Command from a Local Session**



```
[root@localhost ~]# ahgc -H 172.17.10.102 -U admin -P advantech status
[Status of Collectors]
mce         : OK
network     : OK
pcie        : Warning
sensor      : Warning
storage     : Warning
```

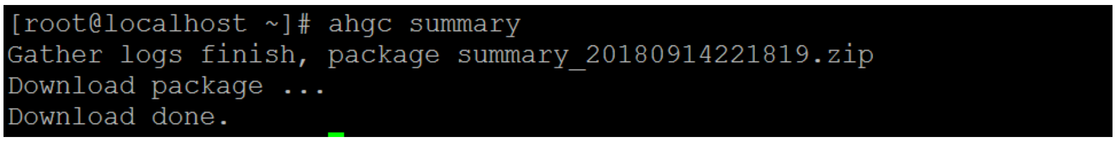
**Figure 23: Status Command from a Remote Session**

### 2.2.2.11 Summary

| Command: summary | | |
|---|---|---|
| **Options** | **Options in short** | **Description** |
| N/A | N/A | Gathers summary information |

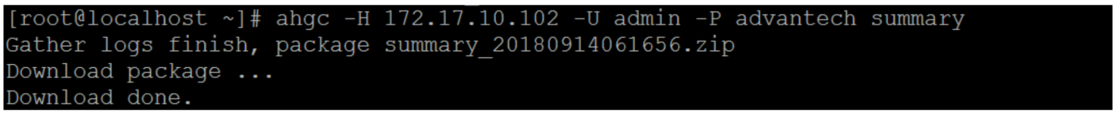
**Table 19: Usage of the Summary Command**

The summary command gathers summary information to a zip file and downloads it to the current working directory. An example of the summary command is shown in Figure 24 (local) and Figure 25 (remote).



```
[root@localhost ~]# ahgc summary
Gather logs finish, package summary_20180914221819.zip
Download package ...
Download done.
```

**Figure 24: Summary Command from a Local Session**



```
[root@localhost ~]# ahgc -H 172.17.10.102 -U admin -P advantech summary
Gather logs finish, package summary_20180914061656.zip
Download package ...
Download done.
```

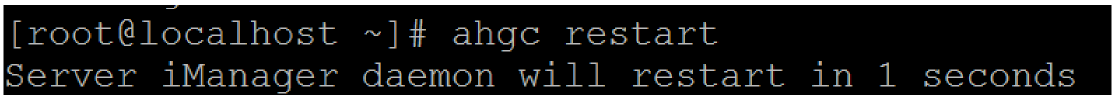
**Figure 25: Summary Command from a Remote Session**

### 2.2.2.12 Restart Command

| Command: restart | | |
|---|---|---|
| **Options** | **Options in short** | **Description** |
| N/A | N/A | Restarts Server iManager daemon |

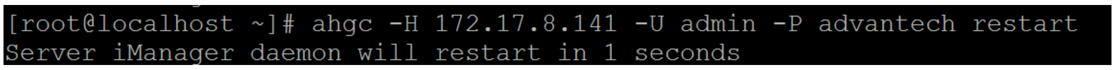
Table 20: Usage of the Restart Command

The restart command restart Server iManager daemon. An example of the restart command is shown in Figure 26 (local session) and Figure 27 (remote session).



```
[root@localhost ~]# ahgc restart
Server iManager daemon will restart in 1 seconds
```

Figure 26: Restart Command from a Local Session



```
[root@localhost ~]# ahgc -H 172.17.8.141 -U admin -P advantech restart
Server iManager daemon will restart in 1 seconds
```

Figure 27: Restart Command from a Remote Session

# 3. Collector and Events

## 3.1 Overview

The collector performs several tasks in Health Guard:

1. Provides a framework for sub-collectors and defines sub-collector behavior.
2. Gathers data from sub-collectors.
3. Checks data status.
4. Provides an API for upstream callers (e.g. REST module).

Information will refresh periodically and the status will also be checked.

## 3.2 Sub-Collectors

Every sub-collector gathers data from the OS or hardware, checks the status, and then reports to the collector.

### 3.2.1 The Sensor Sub-Collector

The sensor sub-collector gets sensor information (e.g., temperature, voltage, and fan speed) and checks the status to generate events. This sub-collector includes the following information:

- Sensor index
- Sensor name
- Sensor value
- Sensor type
- Sensor status

An event will be generated whenever the status of the sensor is abnormal.

### 3.2.2 The Network Sub-Collector

The network sub-collector gets the NIC information and checks the status to generate events. This sub-collector includes the following information:

- Interface index
- Interface name
- Interface type
- MAC address
- IP address
- Link status
- MTU
- Speed
- Duplex
- Auto negotiation
- Driver information
- PCI bus information

- Link statistics

***Note!*** *The interface index starts from the value “2” because the value “1” is for device “lo,” which is skipped.*

***Note!*** *Only wired ports support speed, duplex, and auto negotiation information.*

An event will be generated whenever any of the following occur:

1. Link change: The event reports link up/link down events.
2. IP/speed/duplex/autoneg change: The event reports IP add, IP delete, IP change, and other events.
3. Error counter increment compares with threshold: The error counter increment compares the current error count with the previous error count; if the difference is greater than the network error counter threshold, an event will be generated.
   The default network error counter threshold is 10 (this can be set in the config file).

Error counter includes the following counters:

- rx_errors
- tx_errors
- rx_dropped
- tx_dropped
- collisions
- rx_length_errors
- rx_over_errors
- rx_crc_errors
- rx_frame_errors
- rx_fifio_errors
- rx_missed_errors
- tx_aborted_errors
- tx_carrier_errors
- tx_fifo_errors
- tx_heartbeat_errors
- tx_window_errors

### 3.2.3 The Storage Sub-Collector

The storage sub-collector gets storage information, including that of HDD’s, SSD’s, NVMe’s and virtual drives (hardware RAID), from the LSI RAID controller and SATA controller errors, and then checks the status to generate events. You need to install the “smartctl” utility from the “smartmontools” package, the “StorCLI” utility for LSI RAID information, and the “nvme-cli” utility for NVMe devices information. You can download the “StorCLI” version 1.23.02 from https://docs.broadcom.com/docs/1.23.02_StorCLI.

***Note!*** *The storage sub-collector only supports gathering SATA controller errors on kernel 3.5.0 or later.*

HDDs and SSDs include information of S.M.A.R.T. attributes:

- ID
- Name
- Flag
- Value
- Worst
- Thresh
- Type
- Update
- WhenFailed
- RawValue

NVMes include information of the S.M.A.R.T. attributes:

- Name
- Value

Virtual drives from the LSI RAID controller include the following information:

- Model name of the RAID controller
- Controller ID
- Virtual drive ID
- Virtual drive state
- RAID type
- Size of the virtual drive

SATA controller errors include the following information:

- Value of Serial ATA error register

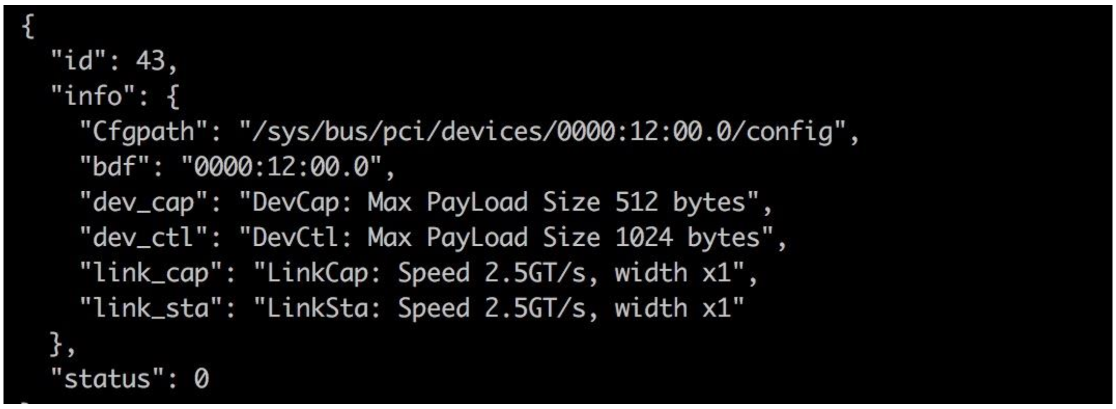
An event will be generated whenever the following occurs:

1. The status of storage changes from “ok” to “critical.” The status of storage is determined by the S.M.A.R.T. status and virtual drive state.
2. The value of the serial ATA error register is not equal to 0.

### 3.2.4 The PCIe Sub-Collector

The PCIe sub-collector gets PCIe/PCI information from the PCIe/PCI device configuration space and checks the status to generate events. This sub-collector includes the following information, as shown in Figure 28:

- Interface index
- Cfgpath (path of PCIe/PCI device file)
- bdf (PCIe/PCI bus/device/function)
- LinkCap (including speed, width)
- LinkSta (including speed, width)
- DevCap (only supports max. payload size)
- DevCtl (only supports max. payload size)



```
{
  "id": 43,
  "info": {
    "Cfgpath": "/sys/bus/pci/devices/0000:12:00.0/config",
    "bdf": "0000:12:00.0",
    "dev_cap": "DevCap: Max PayLoad Size 512 bytes",
    "dev_ctl": "DevCtl: Max PayLoad Size 1024 bytes",
    "link_cap": "LinkCap: Speed 2.5GT/s, width x1",
    "link_sta": "LinkSta: Speed 2.5GT/s, width x1"
  },
  "status": 0
```

**Figure 28: Example of Information from the PCIe/PCI Sub-collector**

An event will be generated whenever any of the following occur:

1. Speed change (without the bridge device type)
2. Normal status error: The event reports that an error bit is set to 1 in the PCIe/PCI status register.
3. AER occurs: The event reports that an error bit is set to 1 in the PCIe AER related register.

***Note!*** *AER is supported only on PCIe devices with an AER mechanism.*

### 3.2.5 The MCE Sub-Collector

The MCE sub-collector gets MCE error information based on the daemon "mcelog." For more about MCE and mcelog, please refer to *Intel® 64 and IA-32 Architectures Software Developer's Manual* and http://www.mcelog.org/.

***Note****! On some new distributions like ubuntu 18.04, mcelog was removed from the OS, this collector will not work on these OS.*

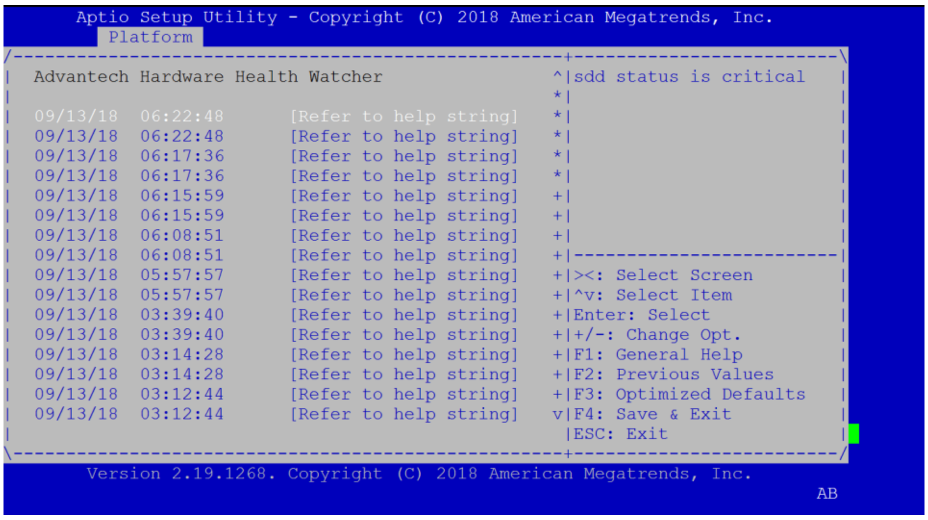
The following is an excerpt of MCE errors in /var/log/mcelog.

```
Hardware event. This is not a software error.
MCE 0
CPU 6 THERMAL EVENT TSC 3cb83afdf0e7e
TIME 1522649766 Mon Apr 2 14:16:06 2018
Processor 6 heated above trip temperature. Throttling enabled.
Please check your system cooling. Performance will be impacted
STATUS 880003c3 MCGSTATUS 0
MCGCAP 1000c14 APICID c SOCKETID 0
PPIN 880003c3
CPUID Vendor Intel Family 6 Model 45
Hardware event. This is not a software error.
MCE 1
CPU 22 THERMAL EVENT TSC 3cb83afdf17f2
TIME 1522649766 Mon Apr 2 14:16:06 2018
Processor 22 heated above trip temperature. Throttling enabled.
Please check your system cooling. Performance will be impacted
STATUS 880003c3 MCGSTATUS 0
MCGCAP 1000c14 APICID d SOCKETID 0
PPIN 88020282
CPUID Vendor Intel Family 6 Model 45
```

An event will be generated when a new MCE error is logged in /var/log/mcelog.

## 3.3 Event

Every sub-collector reports its status to the collector; if an error is detected by a sub-collector, an event will be generated and sent via the notifier, which is described in Chapter 4. If the event is critical, it will be written into the SMBIOS event log and users can check the event from the BIOS setup page as shown in Figure 29:



```
       Aptio Setup Utility - Copyright (C) 2018 American Megatrends, Inc.
          Platform
/--------------------------------------------------------+-------------------------\
|  Advantech Hardware Health Watcher                    ^|sdd status is critical   |
|                                                       *|                         |
|  09/13/18  06:22:48      [Refer to help string]       *|                         |
|  09/13/18  06:22:48      [Refer to help string]       *|                         |
|  09/13/18  06:17:36      [Refer to help string]       *|                         |
|  09/13/18  06:17:36      [Refer to help string]       *|                         |
|  09/13/18  06:15:59      [Refer to help string]       +|                         |
|  09/13/18  06:15:59      [Refer to help string]       +|                         |
|  09/13/18  06:08:51      [Refer to help string]       +|                         |
|  09/13/18  06:08:51      [Refer to help string]       +|-------------------------|
|  09/13/18  05:57:57      [Refer to help string]       +|><: Select Screen        |
|  09/13/18  05:57:57      [Refer to help string]       +|^v: Select Item          |
|  09/13/18  03:39:40      [Refer to help string]       +|Enter: Select            |
|  09/13/18  03:39:40      [Refer to help string]       +|+/-: Change Opt.         |
|  09/13/18  03:14:28      [Refer to help string]       +|F1: General Help         |
|  09/13/18  03:14:28      [Refer to help string]       +|F2: Previous Values      |
|  09/13/18  03:12:44      [Refer to help string]       +|F3: Optimized Defaults   |
|  09/13/18  03:12:44      [Refer to help string]       v|F4: Save & Exit          |
|                                                        |ESC: Exit                |
\--------------------------------------------------------+-------------------------/
        Version 2.19.1268. Copyright (C) 2018 American Megatrends, Inc.
                                                                              AB
```

Figure 29: SMBIOS Event Log

# 4. ALARMING (NOTIFICATION) METHOD

An alarm engine sends notifications via different methods.

***Note!*** *LED is currently unsupported.*

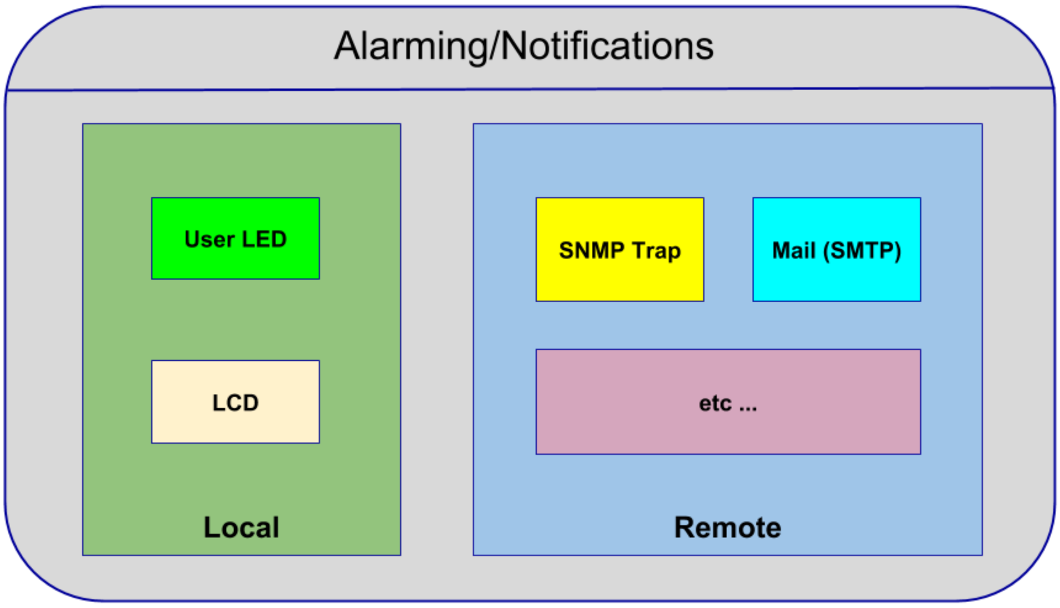


**Figure 30: Alarming Method**

## 4.1 SNMP Trap

If an event is generated by a sub-collector, an SNMP-trap will be sent via the notifier when the SNMP-trap is set to “enable” in the notify section (see Chapter 1.3.5). Traps are defined in MIB files to parse the SNMP-trap on the client.

Health Guard supports the following types of SNMP-trap:

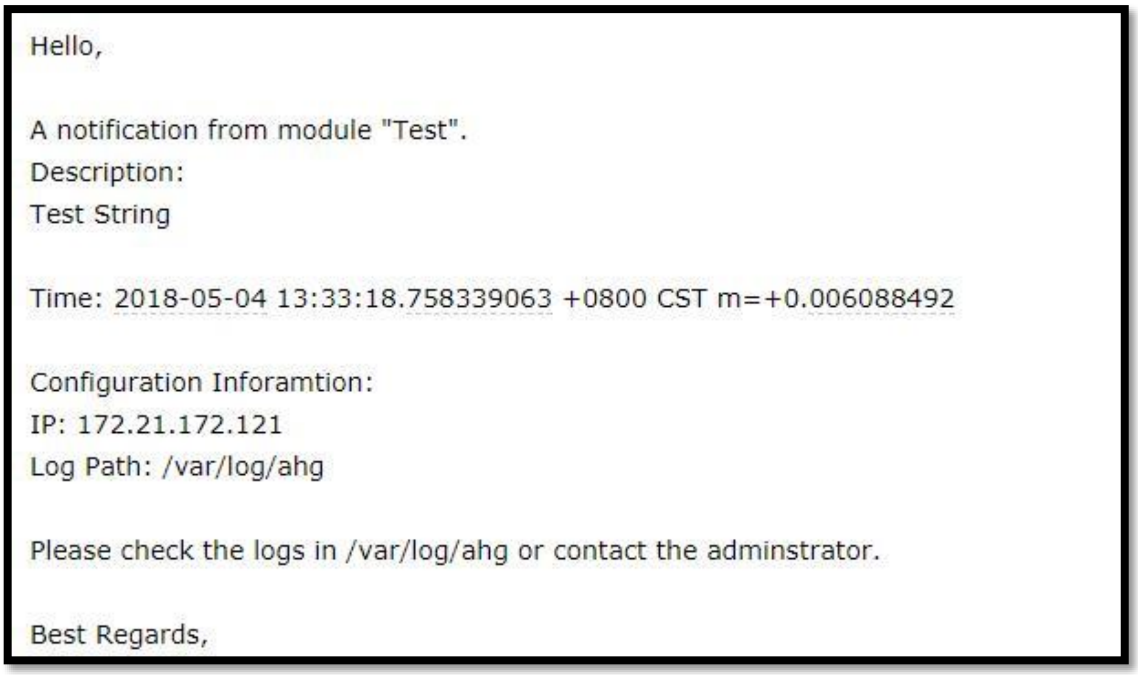
- **Sensor**: A sensor trap signifies that the sensor sub-collector has sent an event. The error could be a power supply error, voltage error, temperature error, or other error. For more details, check the corresponding log file.
- **Network**: A network trap signifies that the network sub-collector has sent an event. The event could be a link down, link up, speed change, or some other error type. The event could also be an error report. For more details, check the log file.
- **Storage**: A storage trap signifies that the storage sub-collector has sent an event. The event could be a device exchange (hot plug), an error report, or some other error. For more details, check the log file.

- **PCIe**: A PCIe/PCI trap signifies that the PCIe sub-collector has sent an event. The event could be an AER error, speed change, link lost, or other event. For more details, check the log file.
- **MCE**: This trap means the MCE sub-collector has detected an MCE error. For more details, check the log file.

## 4.2 SMTP

If an event is generated by a sub-collector, an e-mail will be sent via the notifier if SMTP is set to “enable” in the notify section (see Chapter 1.3.5). You must configure the SMTP-related settings in the config file. For details of the SMTP settings, please refer to Chapter 1.3.6.

Below is an example of a notifier mail based on SMTP:



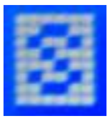
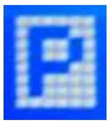
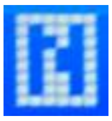
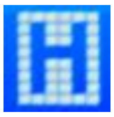
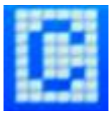
```
Hello,

A notification from module "Test".
Description:
Test String

Time: 2018-05-04 13:33:18.758339063 +0800 CST m=+0.006088492

Configuration Inforamtion:
IP: 172.21.172.121
Log Path: /var/log/ahg

Please check the logs in /var/log/ahg or contact the adminstrator.

Best Regards,
```

**Figure 31: Example of an SMTP-Based Notification Mail**

## 4.3 LCD

If an event is generated by a sub-collector, Health Guard will light up the corresponding icon via the notifier if the LCD is set to “enable” in the notify section (Chapter 1.3.5). Health Guard will switch off the corresponding icon when the status is back to normal.

The icons (6x8) for the sub-collectors are as shown in below:

-  Sensor error
-  PCIe/PCI error
-  Network error
-  HDD (SSD, RAID) error
-  MCE error

# 5. LOG

There are several log files created by the Health Guard components.

TheHealth Guard daemon log will be saved in daemon.log

The Health Guard notifier log will be saved in notifiler.log

The configurations for log module are stored in the config file, which is described in Chapter 1.3.

## 5.1 LOG File Path

By default, the log files are stored in the path /var/log/ahg. This path can be set in the config file.

## 5.2 LOG Level

There are some different log levels in the log files:

- Trace
- Info
- Warn
- Error
- Fatal

Trace is the lowest level and fatal is the highest level.

For example, if the log level is set to "warn", any log message at "info" or "trace" level will not be saved to a log file.

# 6. PROMETHEUS STACK

## 6.1 Introduction

Prometheus is a powerful open-source monitoring and alerting solution. It's widely used especially in OpenStack and Kubernetes. To enable hardware health monitoring and alerting on the server level in those software systems, Health Guard Exporter is introduced here.

The Prometheus ecosystem consists of multiple components, many of which are optional. The four primary components of Prometheus are the Prometheus server, the visualization layer with Grafana, exporters to export metrics, and alert management functions with Prometheus Alertmanager.

- Prometheus server: Scrapes and stores time series data
- Grafana:  Produces dashboards
- Exporter: Collects data from the target host where Health Guard is running, then exposes it to regular "/metrics" endpoints. Details will be addressed in the following sessions.
- Alertmanager: Triggers the alerts

Most Prometheus components are written in Go, making them easy to build and deploy as a static binary.

Advantech provides Health Guard Exporter and the user needs to install the other three components by themselves. In the following sections, we will introduce each primary component of Prometheus and how to deploy them with Health Guard Exporter.

## 6.2 Architecture

The Server iManager Prometheus stack consists of five primary components; the architecture is illustrated as Figure 32.

Server iManager Prometheus stack includes the components below, and will be introduced in the following sections. For usage of Health Guard, please refer to Chapters 1~5.

1. Supports Health Guard daemons in multiple hosts.
2. Health Guard Exporter collects data and alerts from the hosts and exposes metrics.
3. Prometheus server pulls metrics from Health Guard Exporter.
4. Grafana produces the dashboard.
5. Alertmanager handles alerts sent by Prometheus server.

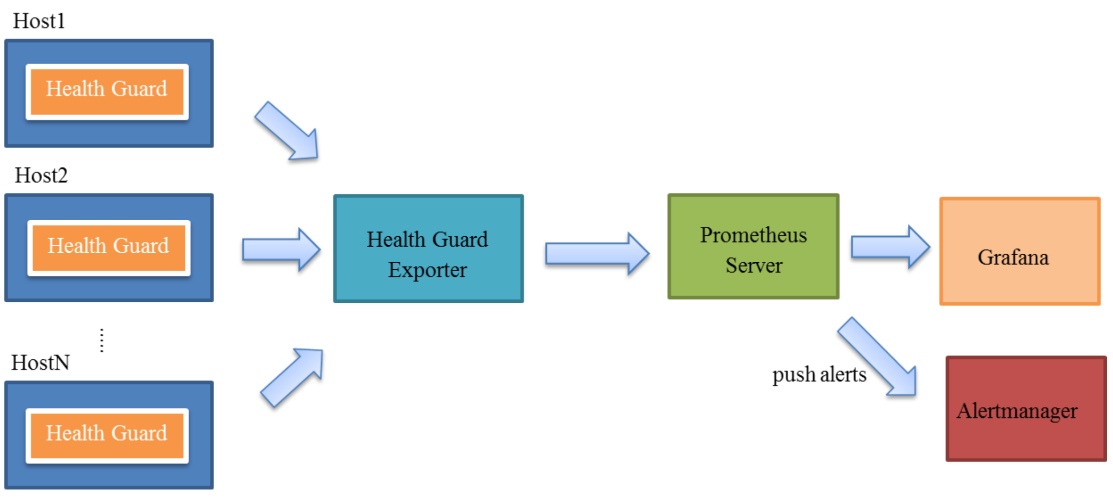


**Figure 32: Architecture of Server iManager Prometheus Stack**

## 6.3 Health Guard Exporter

### 6.3.1 Introduction

Health Guard Exporter is an exporter for Prometheus. It collects data from the target host that Health Guard is running on, then exposes it to the regular "/metrics" endpoint.

### 6.3.2 Configuration

Health Guard Exporter gets the configuration of log files and targets hosts from a config file called "config.ini." For example, "config.ini" is shown in Figure 33:

```
[log]
path     = /var/log/ahg_exporter
level    = info
max-size = 102400000
max-days = 10

[target1]
user     = admin
password = passwd
ip = 192.168.0.1
port = 8087

[target2]
user     = admin
password = passwd
ip = 192.168.0.2
port = 8087
```

**Figure 33: Example Content in Config.ini**

The log section in the example config.ini is followed by the associated keys "path," "level," "max-size" and "max-days." The section name should fix to "log". The keys in log section are defined in the following table.

| KEY | DESCRIPTION |
|---|---|
| Path | Log path (Default: /var/log/ahg_exporter) |
| Level | Log level; currently supported levels are "trace," "info," "warn," "error," and "fatal." |

| | |
|---|---|
| Max-size | The maximize size of the log file (Default: 102,400,000; 100 MB) |
| Max-days | The keep time of the log file (default: 10; 10 days) |

**Table 21: Section Key Definitions for Log**

This example config.ini file also contains 2 sections, “target1” and “target2,” which are followed by the associated keys “user,” “ip,” “password” and “port.” Users can edit the section name to whatever you want. The keys in each section are defined in the following table.

| KEY | DESCRIPTION |
|---|---|
| User | Username used in Health Guard REST authentication.<br>Please refer to your Health Guard configuration file. |
| Password | Password used in Health Guard REST authentication.<br>Please refer to your Health Guard configuration file. |
| IP | The IP address of REST API for Health Guard server daemon. |
| Port | The port that Health Guard server daemon is listening to. |

**Table 22: Section Key Definitions for Target Host**

### 6.3.3 Usage

The syntax of the Health Guard Exporter command is as follows:

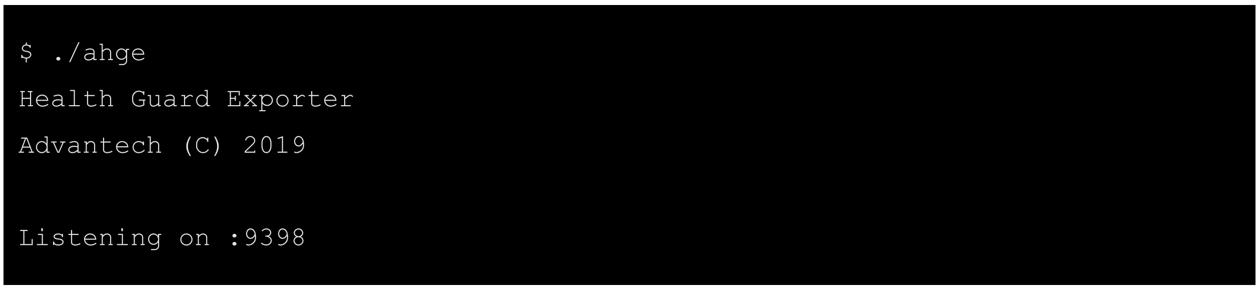
**# ahge [command]**

For details on the usage of each command, please refer to Table 23.

| Command | Description |
|---|---|
| --help | Shows help. |
| --addr string | Sets the http listen address (default port ":9398"). |
| --conf string | Sets the alternative config file. |

**Table 23: Usage of the Health Guard Exporter**

After executing the Health Guard Exporter with default address and config file, the user should see output as Figure 34 indicating that the Health Guard Exporter is running now and exposing metrics on default port 9398:



```
$ ./ahge
Health Guard Exporter
Advantech (C) 2019

Listening on :9398
```

**Figure 34: Running Output of Health Guard Exporter**

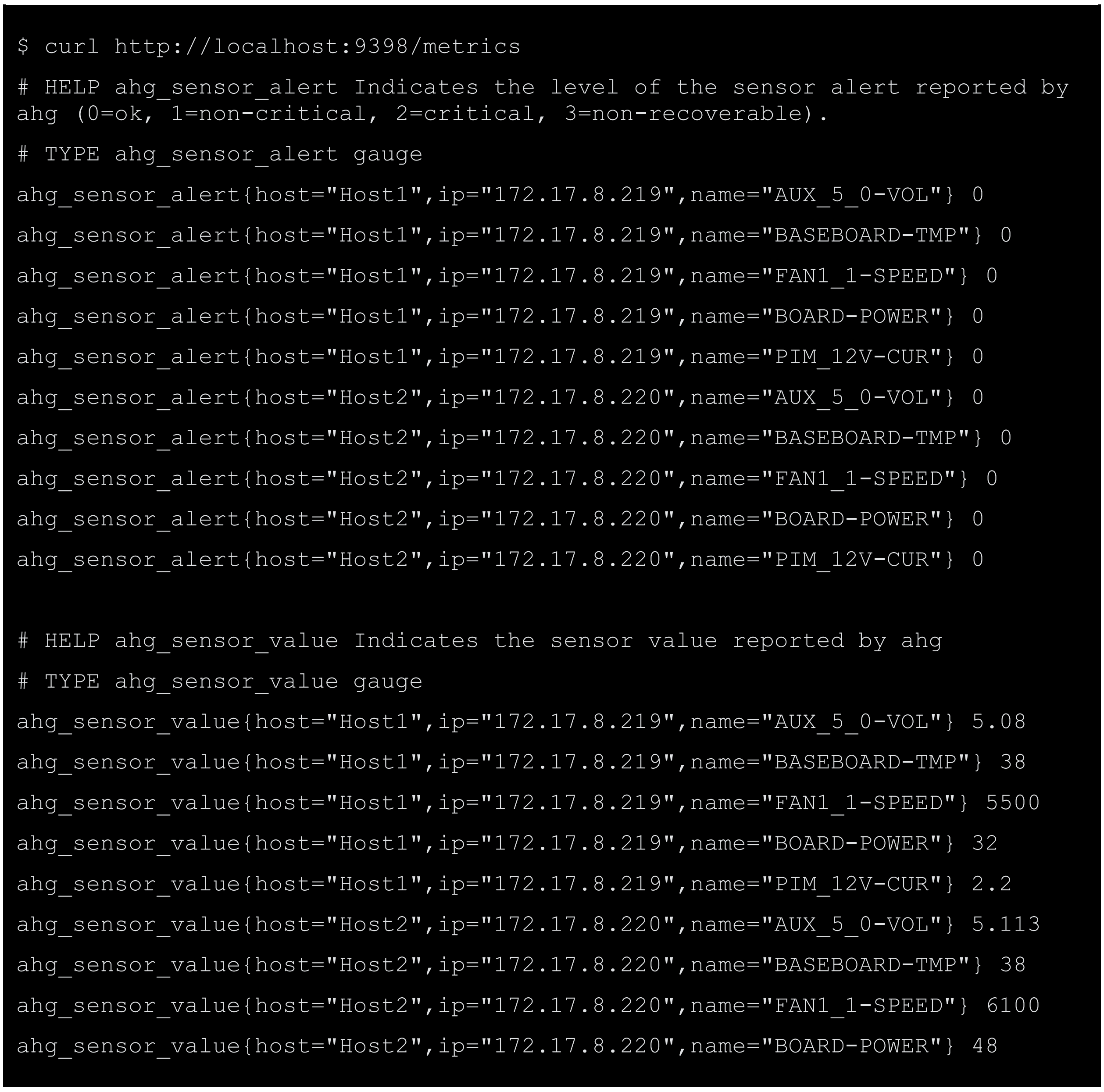
### 6.3.4 Metrics

The Health Guard Exporter exposes the data of Health Guard to metrics for Prometheus.

#### 6.3.4.1 Sensor

This sensor metric is only provided if the sensor sub-collector in Health Guard is enabled on each target host. The Health Guard Exporter exposes sensor data and sensor alerts for all of the sensor types except discrete types and “na” (not available) status from the target host to metrics.

Once Health Guard Exporter is running, the user can verify metrics exported using the command: “curl” with the “/metrics” endpoint, as shown in Figure 35:



```
$ curl http://localhost:9398/metrics
# HELP ahg_sensor_alert Indicates the level of the sensor alert reported by
ahg (0=ok, 1=non-critical, 2=critical, 3=non-recoverable).
# TYPE ahg_sensor_alert gauge
ahg_sensor_alert{host="Host1",ip="172.17.8.219",name="AUX_5_0-VOL"} 0
ahg_sensor_alert{host="Host1",ip="172.17.8.219",name="BASEBOARD-TMP"} 0
ahg_sensor_alert{host="Host1",ip="172.17.8.219",name="FAN1_1-SPEED"} 0
ahg_sensor_alert{host="Host1",ip="172.17.8.219",name="BOARD-POWER"} 0
ahg_sensor_alert{host="Host1",ip="172.17.8.219",name="PIM_12V-CUR"} 0
ahg_sensor_alert{host="Host2",ip="172.17.8.220",name="AUX_5_0-VOL"} 0
ahg_sensor_alert{host="Host2",ip="172.17.8.220",name="BASEBOARD-TMP"} 0
ahg_sensor_alert{host="Host2",ip="172.17.8.220",name="FAN1_1-SPEED"} 0
ahg_sensor_alert{host="Host2",ip="172.17.8.220",name="BOARD-POWER"} 0
ahg_sensor_alert{host="Host2",ip="172.17.8.220",name="PIM_12V-CUR"} 0

# HELP ahg_sensor_value Indicates the sensor value reported by ahg
# TYPE ahg_sensor_value gauge
ahg_sensor_value{host="Host1",ip="172.17.8.219",name="AUX_5_0-VOL"} 5.08
ahg_sensor_value{host="Host1",ip="172.17.8.219",name="BASEBOARD-TMP"} 38
ahg_sensor_value{host="Host1",ip="172.17.8.219",name="FAN1_1-SPEED"} 5500
ahg_sensor_value{host="Host1",ip="172.17.8.219",name="BOARD-POWER"} 32
ahg_sensor_value{host="Host1",ip="172.17.8.219",name="PIM_12V-CUR"} 2.2
ahg_sensor_value{host="Host2",ip="172.17.8.220",name="AUX_5_0-VOL"} 5.113
ahg_sensor_value{host="Host2",ip="172.17.8.220",name="BASEBOARD-TMP"} 38
ahg_sensor_value{host="Host2",ip="172.17.8.220",name="FAN1_1-SPEED"} 6100
ahg_sensor_value{host="Host2",ip="172.17.8.220",name="BOARD-POWER"} 48
```

```
ahg_sensor_value{host="Host2",ip="172.17.8.220",name="PIM_12V-CUR"} 2.8
```

**Figure 35: Health Guard Exporter Sensor Metrics**

The definition of Health Guard Exporter sensor metric prefixed with “ahg_sensor” is as below.

| **Metric** | **Description** |
|---|---|
| ahg_sensor_alert{host="hostname",ip="IP",name="sensor name"} | Indicates the level of the sensor alert reported by ahg without the discrete sensor type and “na” (not available) status (0=ok, 1=non-critical, 2=critical, 3=non-recoverable) |
| ahg_sensor_value{host="hostname",ip="IP",name="sensor name"} | Indicates the sensor value reported by ahg without the discrete sensor type and “na” (not available) status |

**Table 24: Definition of the Health Guard Exporter Sensor Metrics**

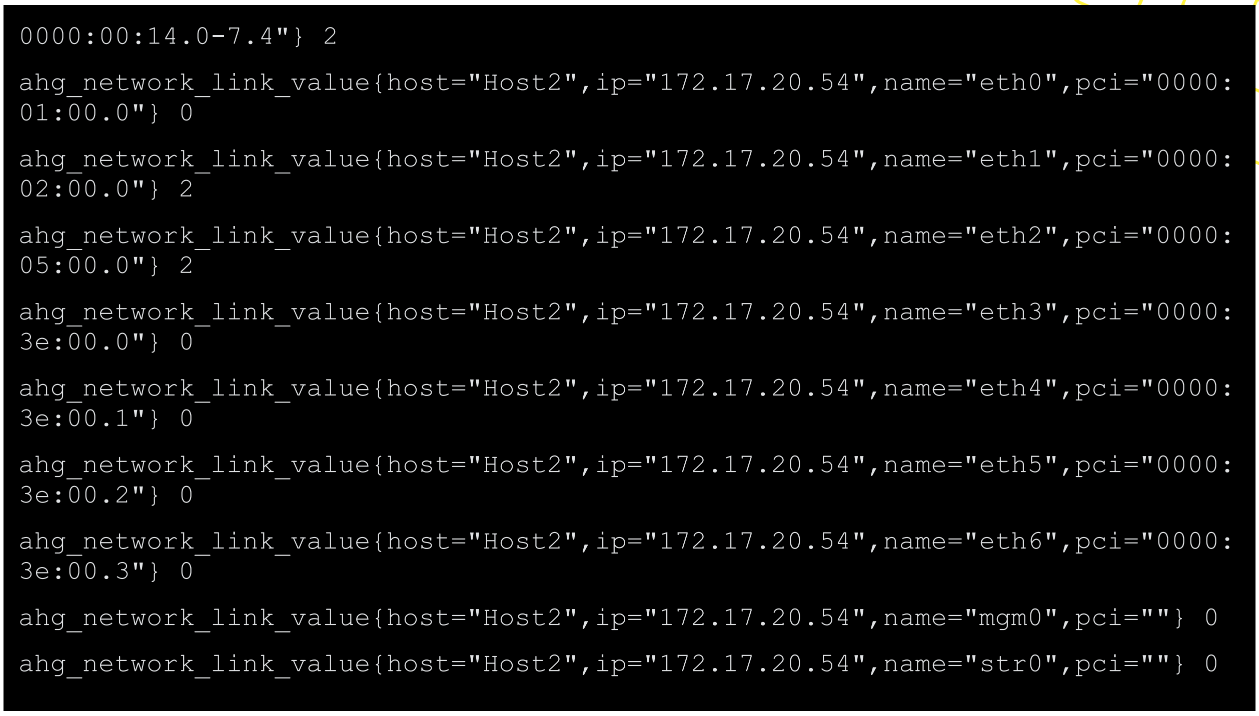
#### 6.3.4.2 Network

This network metric is only provided if the network sub-collector in Health Guard is enabled on each target host. The Health Guard Exporter exposes the network link status & network link alert only for the type “device” or “bond” from the target host to metrics.

Once Health Guard Exporter is running, the user can verify metrics exported using the command: “curl” with the “/metrics” endpoint, as shown in Figure 36:

```
$ curl http://localhost:9398/metrics
# HELP ahg_network_link_alert Indicates the level of the network link alert
reported by ahg (0=ok, 1=non-critical, 2=critical, 3=non-recoverable).
# TYPE ahg_network_link_alert gauge
ahg_network_link_alert{host="Host1",ip="172.17.20.51",name="eth0",pci="0000:
01:00.0"} 0
ahg_network_link_alert{host="Host1",ip="172.17.20.51",name="eth1",pci="0000:
01:00.1"} 0
ahg_network_link_alert{host="Host1",ip="172.17.20.51",name="eth2",pci="0000:
02:00.0"} 0
ahg_network_link_alert{host="Host1",ip="172.17.20.51",name="eth3",pci="0000:
```

```
03:00.0"} 2

ahg_network_link_alert{host="Host1",ip="172.17.20.51",name="eth4",pci="0000:
b5:00.0"} 0

ahg_network_link_alert{host="Host1",ip="172.17.20.51",name="eth5",pci="0000:
b5:00.1"} 2

ahg_network_link_alert{host="Host1",ip="172.17.20.51",name="mgm0",pci=""} 0

ahg_network_link_alert{host="Host1",ip="172.17.20.51",name="str0",pci=""} 0

ahg_network_link_alert{host="Host1",ip="172.17.20.51",name="usb0",pci="usb-
0000:00:14.0-7.4"} 2

ahg_network_link_alert{host="Host2",ip="172.17.20.54",name="eth0",pci="0000:
01:00.0"} 0

ahg_network_link_alert{host="Host2",ip="172.17.20.54",name="eth1",pci="0000:
02:00.0"} 2

ahg_network_link_alert{host="Host2",ip="172.17.20.54",name="eth2",pci="0000:
05:00.0"} 2

ahg_network_link_alert{host="Host2",ip="172.17.20.54",name="eth3",pci="0000:
3e:00.0"} 0

ahg_network_link_alert{host="Host2",ip="172.17.20.54",name="eth4",pci="0000:
3e:00.1"} 0

ahg_network_link_alert{host="Host2",ip="172.17.20.54",name="eth5",pci="0000:
3e:00.2"} 0

ahg_network_link_alert{host="Host2",ip="172.17.20.54",name="eth6",pci="0000:
3e:00.3"} 0

ahg_network_link_alert{host="Host2",ip="172.17.20.54",name="mgm0",pci=""} 0

ahg_network_link_alert{host="Host2",ip="172.17.20.54",name="str0",pci=""} 0

# HELP ahg_network_link_value Indicates the network link status reported by
ahg (0=up, 1=unknown, 2=down)

# TYPE ahg_network_link_value gauge

ahg_network_link_value{host="Host1",ip="172.17.20.51",name="eth0",pci="0000:
01:00.0"} 0

ahg_network_link_value{host="Host1",ip="172.17.20.51",name="eth1",pci="0000:
01:00.1"} 0

ahg_network_link_value{host="Host1",ip="172.17.20.51",name="eth2",pci="0000:
02:00.0"} 0

ahg_network_link_value{host="Host1",ip="172.17.20.51",name="eth3",pci="0000:
03:00.0"} 2

ahg_network_link_value{host="Host1",ip="172.17.20.51",name="eth4",pci="0000:
b5:00.0"} 0

ahg_network_link_value{host="Host1",ip="172.17.20.51",name="eth5",pci="0000:
b5:00.1"} 2

ahg_network_link_value{host="Host1",ip="172.17.20.51",name="mgm0",pci=""} 0

ahg_network_link_value{host="Host1",ip="172.17.20.51",name="str0",pci=""} 0

ahg_network_link_value{host="Host1",ip="172.17.20.51",name="usb0",pci="usb-
```



```
0000:00:14.0-7.4"} 2
ahg_network_link_value{host="Host2",ip="172.17.20.54",name="eth0",pci="0000:
01:00.0"} 0
ahg_network_link_value{host="Host2",ip="172.17.20.54",name="eth1",pci="0000:
02:00.0"} 2
ahg_network_link_value{host="Host2",ip="172.17.20.54",name="eth2",pci="0000:
05:00.0"} 2
ahg_network_link_value{host="Host2",ip="172.17.20.54",name="eth3",pci="0000:
3e:00.0"} 0
ahg_network_link_value{host="Host2",ip="172.17.20.54",name="eth4",pci="0000:
3e:00.1"} 0
ahg_network_link_value{host="Host2",ip="172.17.20.54",name="eth5",pci="0000:
3e:00.2"} 0
ahg_network_link_value{host="Host2",ip="172.17.20.54",name="eth6",pci="0000:
3e:00.3"} 0
ahg_network_link_value{host="Host2",ip="172.17.20.54",name="mgm0",pci=""} 0
ahg_network_link_value{host="Host2",ip="172.17.20.54",name="str0",pci=""} 0
```

**Figure 36: Health Guard Exporter Network Metrics**

The definition of Health Guard Exporter network metric is prefixed with “ahg_ network_link” is as below.

| **Metric** | **Description** |
|---|---|
| ahg_network_link_alert{host="hostname",ip=”IP”,name="network device name",pci="pci information"} | Indicates the level of the network link alert reported by ahg (0=ok, 1=non-critical, 2=critical, 3=non-recoverable) |
| ahg_network_link_value{host=”hostname”,ip="IP",name=" network device name",pci="pci information"} | Indicates the network link status reported by ahg (0=up, 1=unknown, 2=down) |

**Table 25: Definition of the Health Guard Exporter Network Metrics**

### 6.3.4.3 Storage

This storage metric is only provided if the storage sub-collector in Health Guard is enabled on each target host. The Health Guard Exporter exposes the storage status and storage alert from the target host to metrics.

Once Health Guard Exporter is running, the user can verify metrics exported using the command: “curl” with the “/metrics” endpoint, as shown in Figure 37:

```
$ curl http://localhost:9398/metrics
# HELP ahg_storage_alert Indicates the level of the storage alert reported
by ahg (0=ok, 1=non-critical, 2=critical, 3=non-recoverable).
# TYPE ahg_storage_alert gauge
ahg_storage_alert{host="Host1",ip="172.17.8.141",name="sda"} 0
ahg_storage_alert{host="Host2",ip="172.17.8.144",name="sda"} 0
# HELP ahg_storage_value Indicates the storage status reported by ahg (0=ok,
1=critical)
# TYPE ahg_storage_value gauge
ahg_storage_value{host="Host1",ip="172.17.8.141",name="sda"} 0
ahg_storage_value{host="Host2",ip="172.17.8.144",name="sda"} 0
```

**Figure 37: Health Guard Exporter Storage Metrics**

The definition of Health Guard Exporter storage metric is prefixed with “ahg_storage” is as below.

| **Metric** | **Description** |
|---|---|
| ahg_storage_alert{host=”hostname”,ip="IP",name=" storage device name"} | Indicates the level of the storage alert reported by ahg (0=ok, 1=non-critical, 2=critical, 3=non-recoverable) |
| ahg_storage_value{ host=”hostname”,ip="IP",name=" storage device name"} | Indicates the storage status reported by ahg (0=ok, 1=non-critical) |

**Table 26: Definition of the Health Guard Exporter Storage Metrics**

#### 6.3.4.4 PCI

This PCI metric is only provided if the PCIe sub-collector in Health Guard is enabled on each target host. The Health Guard Exporter exposes the PCI status and PCI alert from the target. Once Health Guard Exporter is running, the user can verify exported metrics by using the command: “curl” with the “/metrics” endpoint, as shown in Figure 38:

```
$ curl http://localhost:9398/metrics
# HELP ahg_pci_alert Indicates the level of the pci alert reported by ahg
(0=ok, 1=non-critical, 2=critical, 3=non-recoverable).
# TYPE ahg_pci_alert gauge
ahg_pci_alert{host="Host1",ip="172.17.8.141",pci="0000:00:00.0"} 0
ahg_pci_alert{host="Host1",ip="172.17.8.141",pci="0000:00:04.0"} 0
ahg_pci_alert{host="Host1",ip="172.17.8.141",pci="0000:00:04.1"} 0
ahg_pci_alert{host="Host2",ip="172.17.8.144",pci="0000:00:00.0"} 0
ahg_pci_alert{host="Host2",ip="172.17.8.144",pci="0000:00:04.0"} 0
ahg_pci_alert{host="Host2",ip="172.17.8.144",pci="0000:00:04.1"} 0
# HELP ahg_pci_value Indicates the pci status value reported by ahg (0=ok,
1=correctable error, 2=uncorrectable error, 3=general error, 4=speed error)
# TYPE ahg_pci_value gauge
ahg_pci_value{host="Host1",ip="172.17.8.141",pci="0000:00:00.0"} 0
ahg_pci_value{host="Host1",ip="172.17.8.141",pci="0000:00:04.0"} 0
ahg_pci_value{host="Host1",ip="172.17.8.141",pci="0000:00:04.1"} 0
ahg_pci_value{host="Host2",ip="172.17.8.144",pci="0000:00:00.0"} 0
ahg_pci_value{host="Host2",ip="172.17.8.144",pci="0000:00:04.0"} 0
ahg_pci_value{host="Host2",ip="172.17.8.144",pci="0000:00:04.1"} 0
```

**Figure 38: Health Guard Exporter Pci Metrics**

The definition of Health Guard Exporter pci metric is prefixed with “ahg_pci” is as below.

| Metric | Description |
|---|---|
| ahg_pci_alert{host=”hostname”,ip="IP",pci=" pci bus, device, function number"} | Indicates the level of the pci alert reported by ahg (0=ok, 1=non-critical, 2=critical, 3=non-recoverable) |
| ahg_pci_value{host=”hostname”,ip="IP", pci=" pci bus, device, function number"} | Indicates the pci status reported by ahg (0=ok, 1=correctable error, 2=uncorrectable error, 3=general error, 4=speed error) |

**Table 27: Definition of the Health Guard Exporter Pci Metrics**

## 6.4 Prometheus Server

### 6.4.1 Introduction

Prometheus is an open-source software application used for event monitoring and alerting. It records real-time metrics in a time series database (allowing for high dimensionality) built using a HTTP pull model, with flexible queries and real-time alerting. The project is written in Go and licensed under the Apache 2 License, with source code available on GitHub. It is a graduated project of the Cloud Native Computing Foundation, along with Kubernetes and Envoy.

### 6.4.2 Installation

The steps for installing and setting up the Prometheus server are as below.

1. Download the latest Prometheus precompiled binary:

   $ wget
   https://github.com/prometheus/prometheus/releases/download/v2.11.1/prometheus-2.11.1.linux-amd64.tar.gz

2. Extract the tar archive:

   $ tar xzvf prometheus-2.11.1.linux-amd64.tar.gz

3. Enter into the Prometheus directory:

   $ cd prometheus-2.11.1.linux-amd64

4. Add Health Guard Exporter in the configuration file of Prometheus server:

   $ vim prometheus.yml

   Add the setting of the Health Guard Exporter in red into “prometheus.yml.”

```
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries
scraped from this config.
  - job_name: 'prometheus'

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.
    static_configs:
    - targets: ['localhost:9090']
  - job_name: 'HealthGuard Exporter resources'
    scrape_interval: 10s
    static_configs:
    - targets: ['172.17.8.219:9398']
```

**Figure 39: Configuration for Health Guard Exporter**

In the example above, Prometheus server is set at default port 9090 and the Health Guard Exporter port is at default 9398. Prometheus server scraps Health Guard Exporter every 10 seconds for metrics. For further information of the configuration in Prometheus server, please refer to: https://prometheus.io/docs/prometheus/latest/configuration/configuration/.

5. Add Alertmanager settings in the configuration file of Prometheus server. Alertmanager is introduced in the section 0 :

   $ vim prometheus.yml

   Add the setting of Alertmanager in red below into “prometheus.yml.”

```
# Alertmanager configuration
alerting:
  alertmanagers:
  - static_configs:
    - targets: ['172.17.8.219:9093']
      # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global
'evaluation_interval'.
rule_files:
  - 'alert.rules'
  # - "first_rules.yml"
  # - "second_rules.yml"
```

**Figure 40: Config for Alertmanager**

In the example above, Alertmanager is set at default port 9093 and it loads the rules from “alert.rules.”

6. Define the alerting rules in “alert.rules.”
   For further information of “alerting rules,” please refer to
   https://prometheus.io/docs/prometheus/latest/configuration/alerting_rules/:

   $ vim alert.rules

   Add the following setting of alerting rules in the file “alert.rules.”

```
groups:
- name: ahg-alert
  rules:
  - alert: non_critical_ahg_sensor_alert
    expr: ahg_sensor_alert == 1
    for: 1m
    labels:
      alert_level: non_critical
    annotations:
      summary: "{{$labels.host}}: Non-critical alert detected"
      description: "{{$labels.host}} {{$labels.name}} current alert value
is: {{ $value }}"
  - alert: critical_ahg_sensor_alert
    expr: ahg_sensor_alert == 2
    for: 1m
    labels:
      alert_level: critical
    annotations:
      summary: "{{$labels.host}}: Critical alert detected"
      description: "{{$labels.host}} {{$labels.name}} current alert value
is: {{ $value }}
```

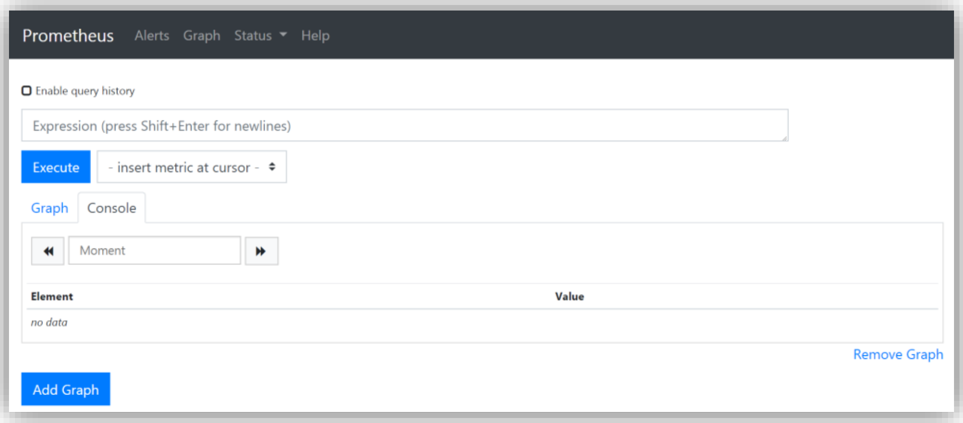
**Figure 41: The setting of the alerting rules**

In the example above, alert rules are set for any instance of alert level 1 and 2. There is a sample file “alert.rules” in Health Guard Exporter release package for reference.

7. Run Prometheus server:

```
$ ./prometheus
```

After running the Prometheus server, the user can see the web page via URL "http://localhost:9090" as shown in Figure 42:



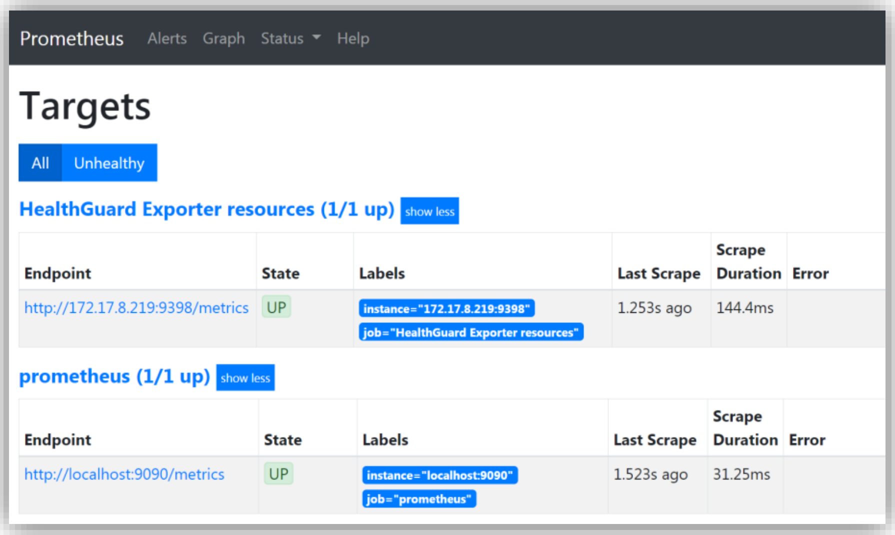
| Element | Value |
|---|---|
| no data | |

**Figure 42: Web page of Prometheus server**

8. Run Health Guard Exporter:

```
$ ./ahge
```

After executing the Health Guard Exporter, the user can confirm the state of Health Guard Exporter from the web page via URL "http://localhost:9090/targets" as shown in Figure 43:



HealthGuard Exporter resources (1/1 up)

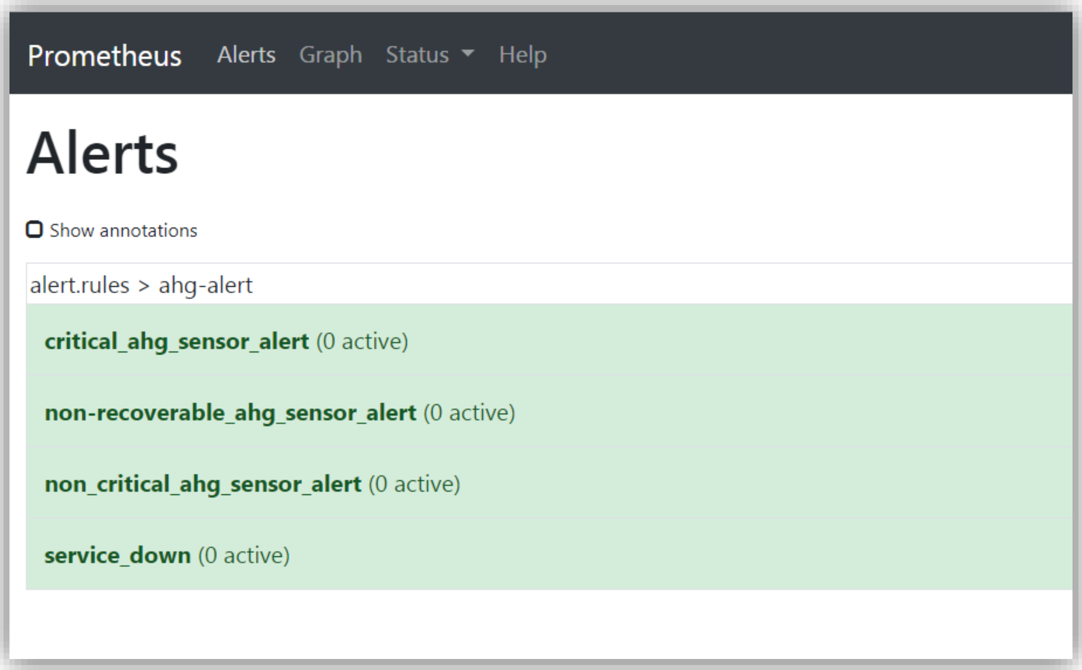
| Endpoint | State | Labels | Last Scrape | Scrape Duration | Error |
|---|---|---|---|---|---|
| http://172.17.8.219:9398/metrics | UP | instance="172.17.8.219:9398" job="HealthGuard Exporter resources" | 1.253s ago | 144.4ms | |

prometheus (1/1 up)

| Endpoint | State | Labels | Last Scrape | Scrape Duration | Error |
|---|---|---|---|---|---|
| http://localhost:9090/metrics | UP | instance="localhost:9090" job="prometheus" | 1.523s ago | 31.25ms | |

**Figure 43: Web interface of Target page in Prometheus server**

The user can see the state of alerts in the Alerts page via URL “http://localhost:9090/alerts” as shown in Figure 44:



Alerts

Show annotations

alert.rules > ahg-alert

**critical_ahg_sensor_alert** (0 active)

**non-recoverable_ahg_sensor_alert** (0 active)

**non_critical_ahg_sensor_alert** (0 active)

**service_down** (0 active)

**Figure 44: Web interface of alert page in Prometheus server**

## 6.5 Grafana Dashboard

### 6.5.1 Introduction

Grafana is open-source software for monitoring and analysis. One of its major characteristics is it supports many different data sources, from popular CloudWatch, Elasticsearch, Graphite, influxDB, Prometheus, and many others. Its range is very extensive. Administrators or operators do not have to use a number of different monitoring software due to the limitation of different information sources.

### 6.5.2 Installation

The steps for installing and setting up the Grafana dashboard are shown below.

1. Download the latest Grafana dashboard for different Linux distributions:
   Ubuntu & Debian:
   ```
   $ wget https://dl.grafana.com/oss/release/grafana_6.3.2_amd64.deb
   $ sudo dpkg -i grafana_6.3.2_amd64.deb
   ```
   Redhat & Centos:
   ```
   $ wget https://dl.grafana.com/oss/release/grafana-6.3.2-1.x86_64.rpm
   $ sudo yum localinstall grafana-6.3.2-1.x86_64.rpm
   ```

2. Run Grafana dashboard:
   ```
   $ systemctl start grafana-server
   ```

After executing the Grafana dashboard, user can see the login page via URL "[http://localhost:3000](http://localhost:3000)" as shown in Figure 45. After logging in with default admin:admin creds, you will be asked to change the password upon successful login or you can skip it.

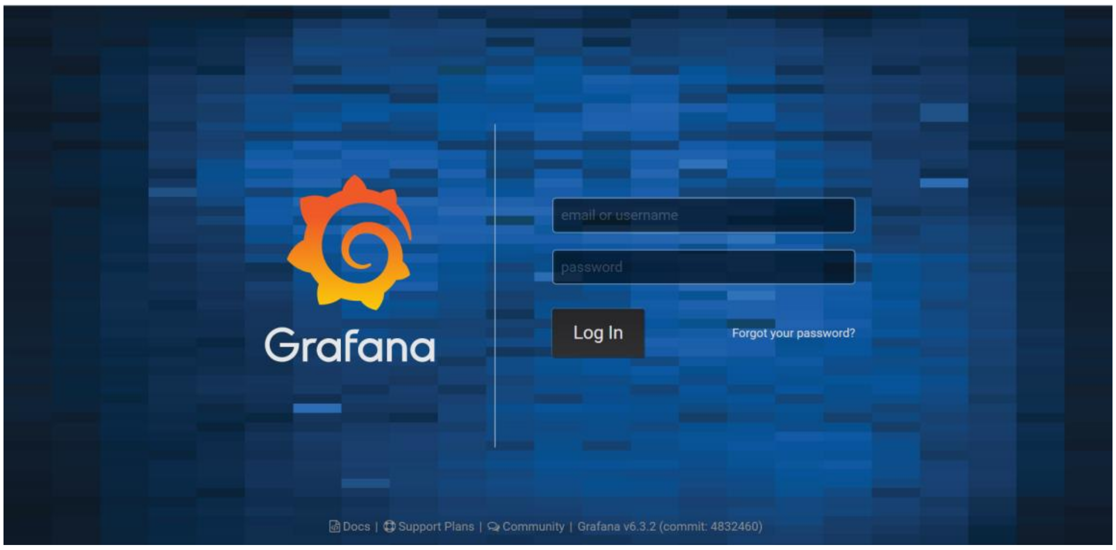
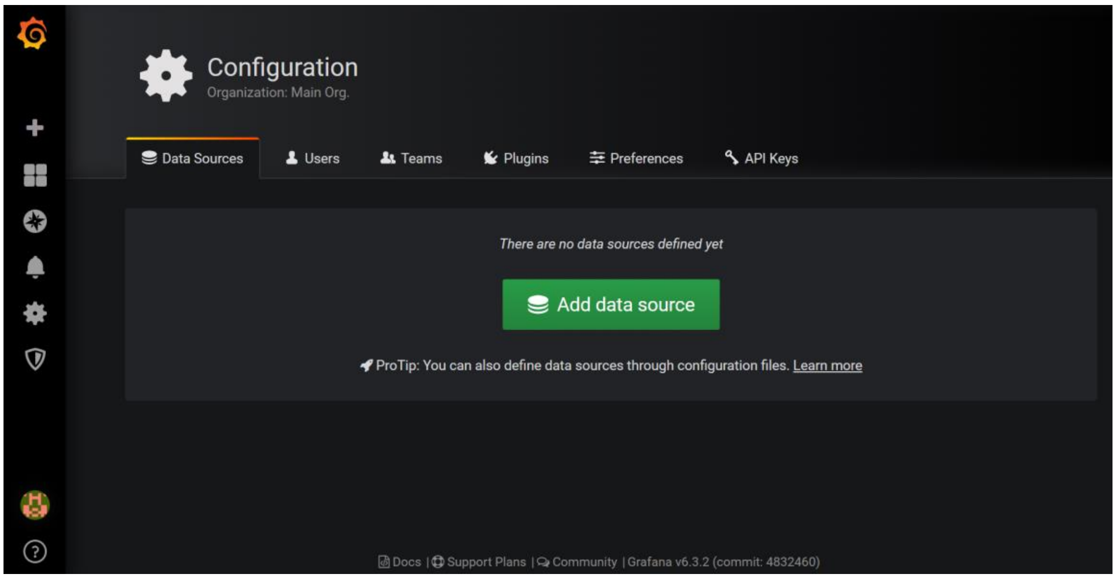

Grafana
email or username
password
Log In
Forgot your password?
Docs | Support Plans | Community | Grafana v6.3.2 (commit: 4832460)

**Figure 45: Login page of Grafana Dashboard**

3. Once you log in, click on “Add Data Source” to add your own Prometheus Server as the source as shown in Figure 46, filling in the configuration and then clicking “Save and Test” as shown in Figure 47.

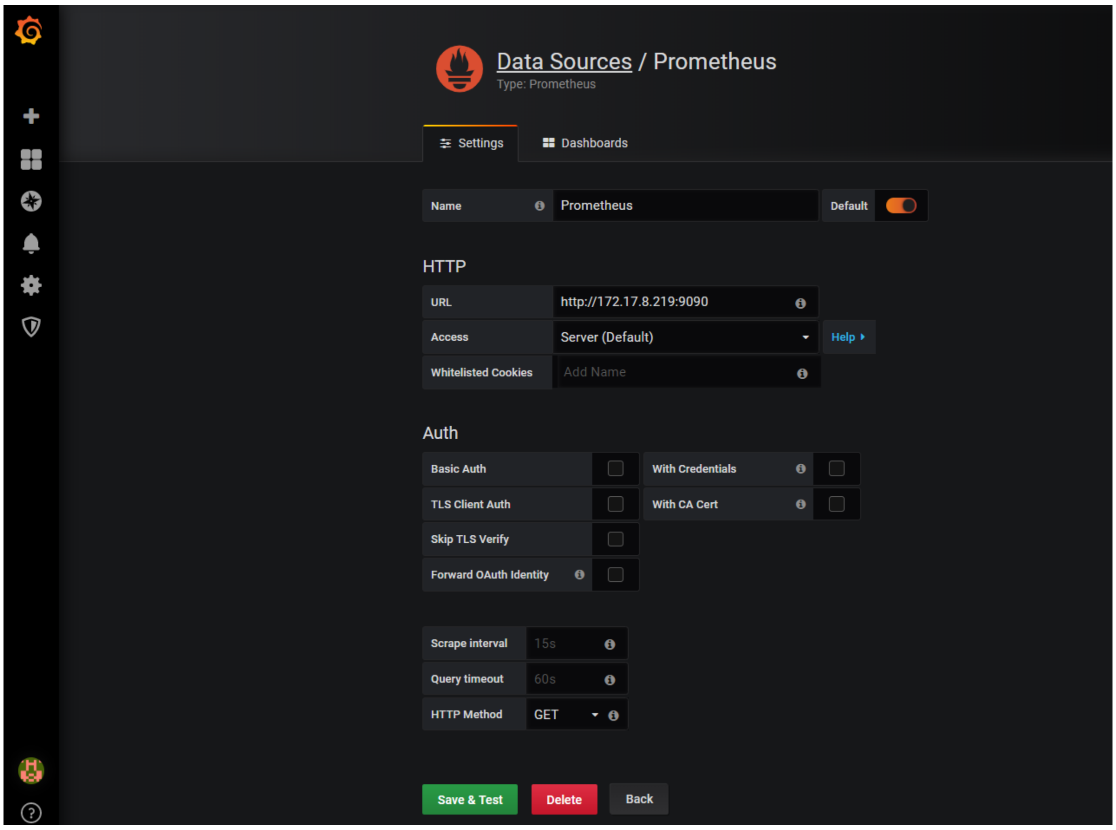


There are no data sources defined yet

ProTip: You can also define data sources through configuration files. Learn more

Docs | Support Plans | Community | Grafana v6.3.2 (commit: 4832460)

Figure 46: Add data source page of Grafana Dashboard



Name: Prometheus
Default: on

HTTP

URL: http://172.17.8.219:9090
Access: Server (Default)
Help
Whitelisted Cookies: Add Name

Auth

| | | | |
|---|---|---|---|
| Basic Auth | ☐ | With Credentials | ☐ |
| TLS Client Auth | ☐ | With CA Cert | ☐ |
| Skip TLS Verify | ☐ | | |
| Forward OAuth Identity | ☐ | | |

Scrape interval: 15s
Query timeout: 60s
HTTP Method: GET

Figure 47: Add data source setting page of Grafana Dashboard

4. Proceed to create the dashboard as shown in Figure 48.

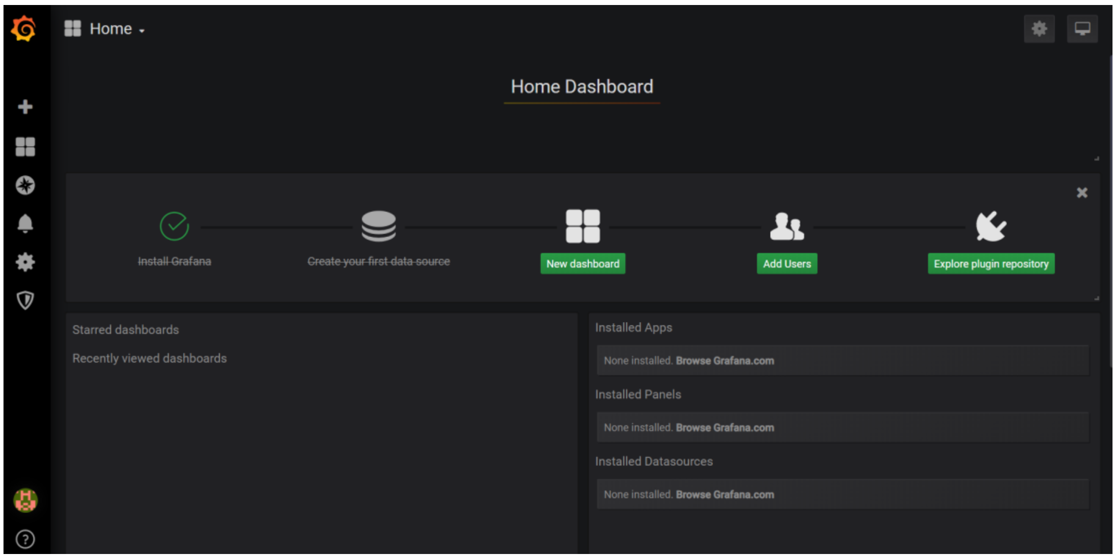

Home
Home Dashboard
~~Install Grafana~~
~~Create your first data source~~
New dashboard
Add Users
Explore plugin repository
Starred dashboards
Recently viewed dashboards
Installed Apps
None installed. Browse Grafana.com
Installed Panels
None installed. Browse Grafana.com
Installed Datasources
None installed. Browse Grafana.com

**Figure 48: New dashboard page of Grafana Dashboard**

5. You will end up with a new panel in dashboard as shown in Figure 49 and corresponding setting page in Figure 50. For further information of “Using Prometheus in Grafana,” please refer to https://grafana.com/docs/features/datasources/prometheus/

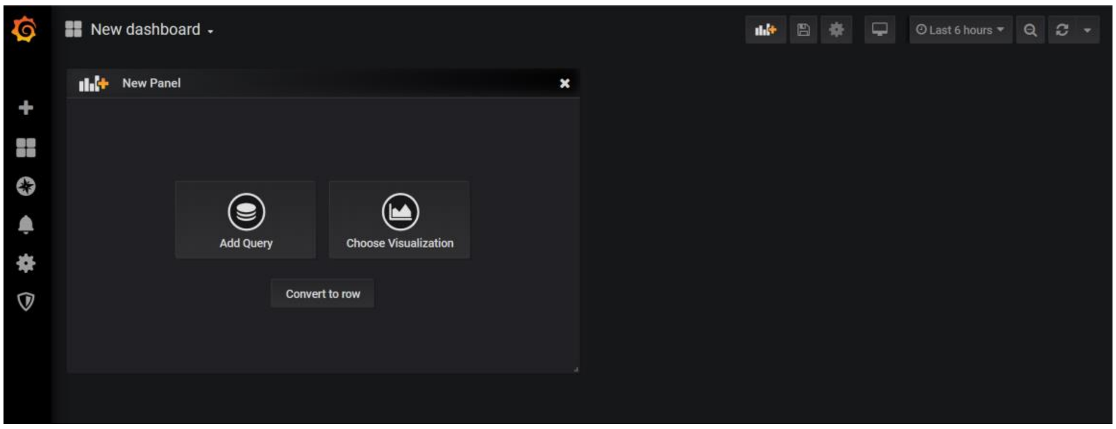


Figure 49: New dashboard of Grafana Dashboard

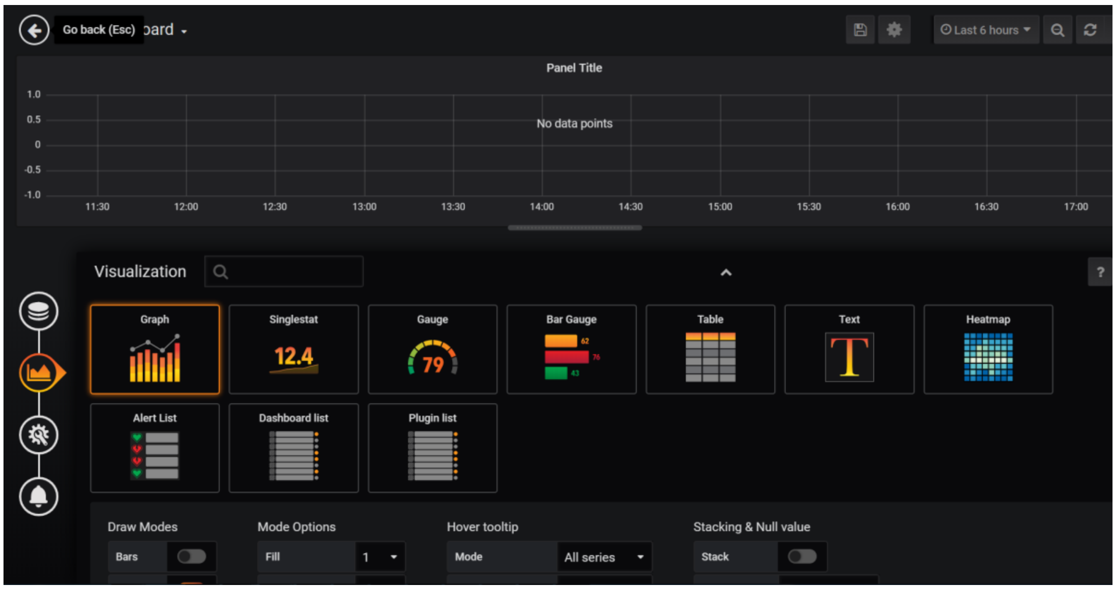
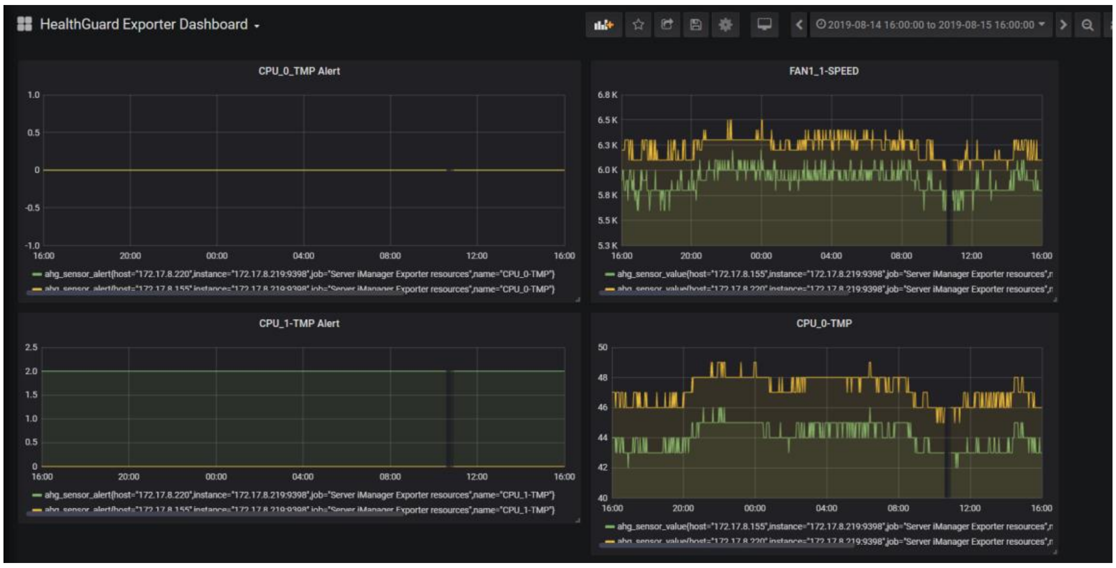


Figure 50: Panel setting page of Grafana Dashboard

6. Figure 51  is the example dashboard of Health Guard Exporter with CPU temp alert and fan speed panels.


HealthGuard Exporter Dashboard
2019-08-14 16:00:00 to 2019-08-15 16:00:00
CPU_0_TMP Alert
FAN1_1-SPEED
CPU_1-TMP Alert
CPU_0-TMP

ahg_sensor_alert{host="172.17.8.220",instance="172.17.8.219:9398",job="Server iManager Exporter resources",name="CPU_0-TMP"}
ahg_sensor_alert{host="172.17.8.155",instance="172.17.8.219:9398",job="Server iManager Exporter resources",name="CPU_0-TMP"}

ahg_sensor_value{host="172.17.8.155",instance="172.17.8.219:9398",job="Server iManager Exporter resources",n
ahg_sensor_value{host="172.17.8.220",instance="172.17.8.219:9398",job="Server iManager Exporter resources",n

ahg_sensor_alert{host="172.17.8.220",instance="172.17.8.219:9398",job="Server iManager Exporter resources",name="CPU_1-TMP"}
ahg_sensor_alert{host="172.17.8.155",instance="172.17.8.219:9398",job="Server iManager Exporter resources",name="CPU_1-TMP"}

ahg_sensor_value{host="172.17.8.155",instance="172.17.8.219:9398",job="Server iManager Exporter resources",n
ahg_sensor_value{host="172.17.8.220",instance="172.17.8.219:9398",job="Server iManager Exporter resources",n

**Figure 51: Example dashboard**

## 6.6 Alertmanager

### 6.6.1 Introduction

The Alertmanager handles alerts sent by client applications such as the Prometheus server. It takes care of deduplicating, grouping, and routing them to the correct receiver integration such as email, PagerDuty, or OpsGenie. It also takes care of silencing and inhibition of alerts.

### 6.6.2 Installation

The steps of installing and setting up the Alertmanager are as below.

1. Download the latest Alertmanager precompiled binary:

   ```
   $ wget
   https://github.com/prometheus/alertmanager/releases/download/v0.18.0/alertmanager-0.18.0.linux-amd64.tar.gz
   ```

2. Extract the tar archive:

   ```
   $ tar xzvf alertmanager-0.18.0.linux-amd64.tar.gz
   ```

3. Enter into the Alertmanager directory:

   ```
   $ cd alertmanager-0.18.0.linux-amd64
   ```

4. Add gmail setting in the configuration file of Alertmanager:

   ```
   $ vim alertmanager.yml
   ```

   Add the gmail setting in red text into “alertmanager.yml.”

```
global:
  resolve_timeout: 5m
  smtp_smarthost: 'smtp.gmail.com:587'
  smtp_from: 'testaccount@gmail.com'
  smtp_auth_username: 'testaccount@gmail.com'
  smtp_auth_password: '*****'

route:
  group_by: ['alertname']
```

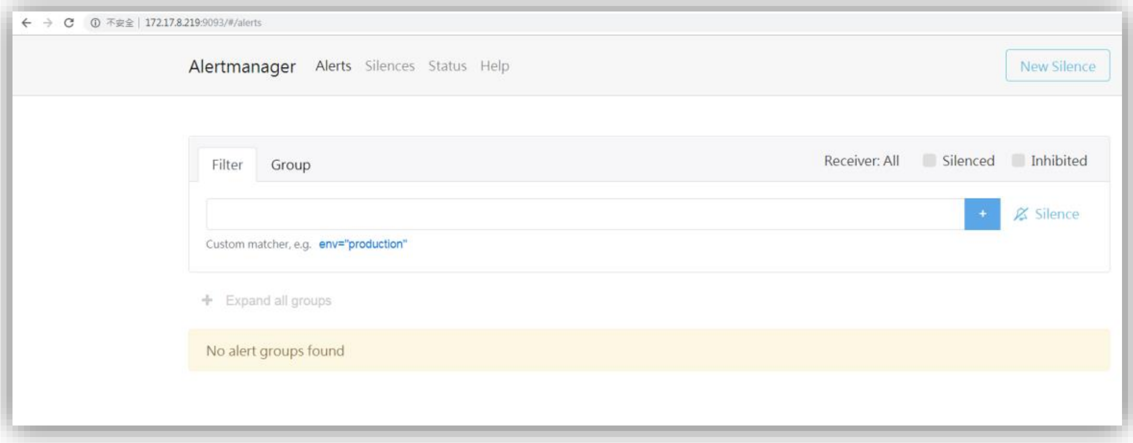
```
  group_wait: 10s
  group_interval: 10s
  repeat_interval: 1h
  receiver: 'gmail'
receivers:
- name: 'gmail'
  email_configs:
  - to: 'testaccount@gmail.com'
    send_resolved: true
inhibit_rules:
  - source_match:
      severity: 'critical'
    target_match:
      severity: 'warning'
    equal: ['alertname', 'dev', 'instance']
```

**Figure 52: Configuration for Alertmanager**

See the example given above to send the notifications to specific gmail accounts via Alertmanager.

9. Run Alertmanager:
   $ ./alertmanager

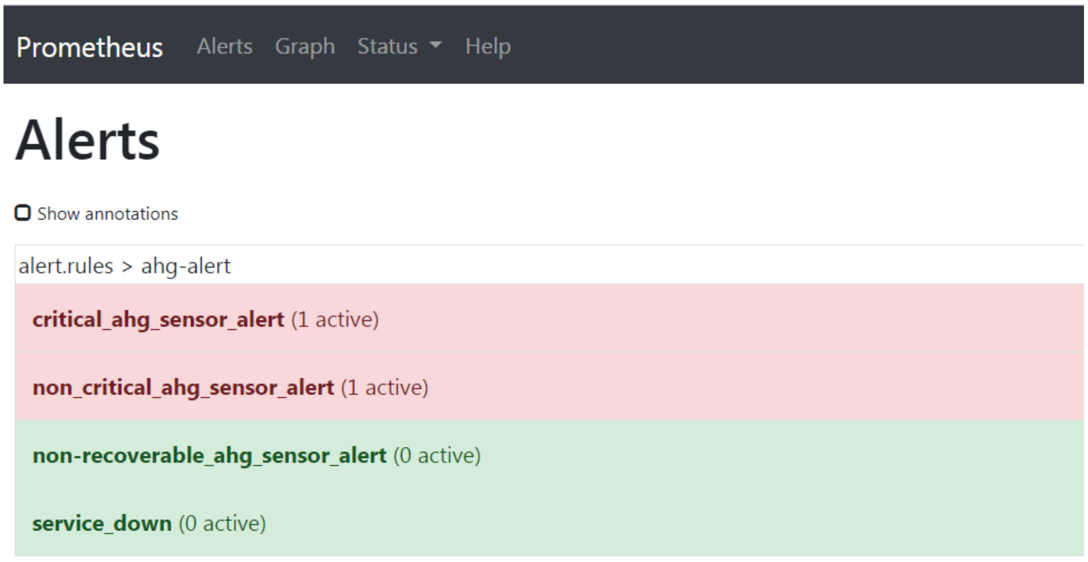
After executing the Alertmanager, the user can see the following web page via URL "http://localhost:9093" as shown in Figure 53:



No alert groups found

**Figure 53: Web page of Alertmanager**

User can see the alert status via URL "http://localhost:9090/alerts" and "http://localhost:9093" as shown in Figure 54-Figure 55:



alert.rules > ahg-alert

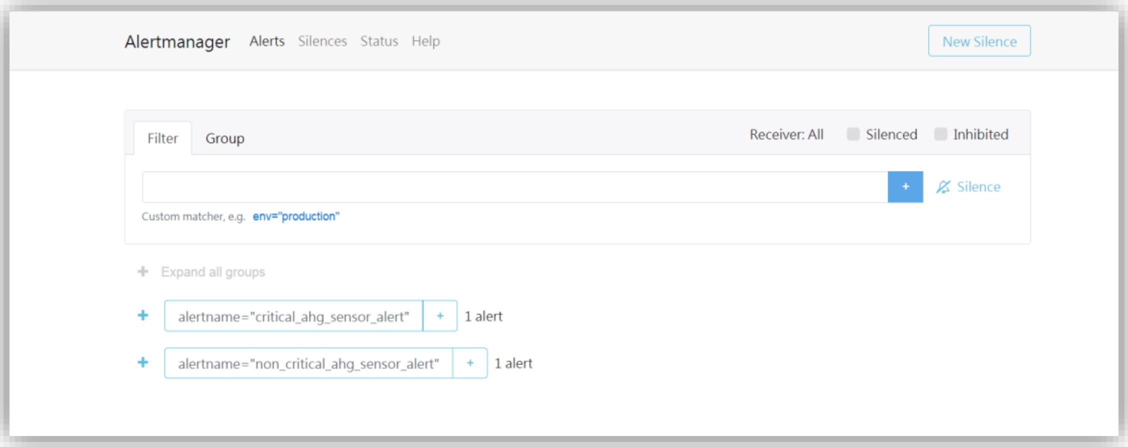
**critical_ahg_sensor_alert** (1 active)

**non_critical_ahg_sensor_alert** (1 active)

**non-recoverable_ahg_sensor_alert** (0 active)

**service_down** (0 active)

**Figure 54: Alerts page of Prometheus server**

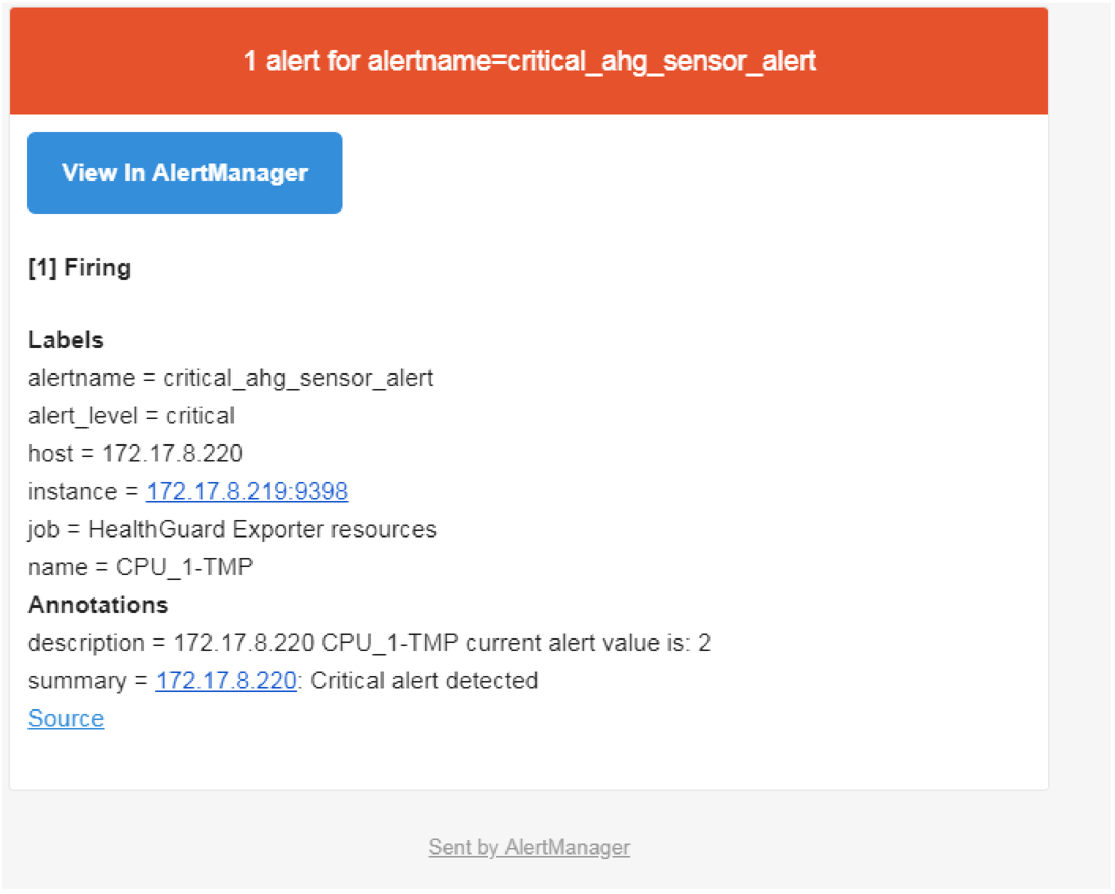


alertname="critical_ahg_sensor_alert" 1 alert

alertname="non_critical_ahg_sensor_alert" 1 alert

**Figure 55: Web page of Alertmanager**

Users can also get notifications from gmail as shown in Figure 56:



1 alert for alertname=critical_ahg_sensor_alert

**[1] Firing**

**Labels**
alertname = critical_ahg_sensor_alert
alert_level = critical
host = 172.17.8.220
instance = 172.17.8.219:9398
job = HealthGuard Exporter resources
name = CPU_1-TMP
**Annotations**
description = 172.17.8.220 CPU_1-TMP current alert value is: 2
summary = 172.17.8.220: Critical alert detected
Source

Sent by AlertManager

**Figure 56: Notification via gmail**

# A. Appendix: Troubleshooting

- *In the CLI utility (ahgc), the default value of the port for REST session is "8087"; please specify the port after the port has been modified. Otherwise, the REST session will fail.*
- *The MCE feature is based on the utility "mcelog"; please ensure that mcelog is installed on your system. Execute mcelog with the command "mcelog --daemon" to output the log to the path /var/log/mcelog.*
  ***Note**! On some new distributions like ubuntu 18.04, mcelog was removed from the OS, this mce collector will not works on these OS.*
- *The storage feature is based on the utilities "smartctl", "StorCLI" and "nvme-cli."*