GPU TECHNOLOGY CONFERENCE
April 4-7, 2016 | Silicon Valley

DATA CENTER GPU MANAGER (DCGM)

Brent Stolle and Rajat Phull, 4/5/2016

PRESENTED BY
NVIDIA.

DATA CENTER INFRASTRUCTURE CHALLENGES

Resource Availability & Uptime

Under-utilized Resources & Efficiency

Administrative Overhead

DATA CENTER GPU MANAGER

Existing Tools

Device Management

Per GPU Configuration & Monitoring

- Device Identification
- Configuration & Monitoring
- Clock Management

All GPUs Supported

DCGM

Tesla GPUs Only

Active Diagnostics and Health Checks

Increases Reliability

Policy & Configuration Management

Lower Admin overhead

Enhanced Clock & Power management

Increases Efficiency

Stateful Group Operations

Maintains historical info

Easy of Use

NVIDIA DATA CENTER GPU MANAGER (DCGM)

Comprehensive GPU Management for Accelerated Data Center

- Maximize GPU Reliability & Uptime
- Streamline GPU Administration & TCO
- Boost Performance & Resource Efficiency

Maximize GPU Reliability & Availability

Active Health Monitoring & Analysis

Comprehensive Diagnostics

Active Health Monitoring & Analysis

NON INVASIVE

Performed during job execution

Overall health for the GPU subsystems (PCIe, SM, MCU, PMU, Inforom, Power and thermal system)

Create Group

```
dcgmi group --create all_gpus_grp --default
Successfully created group "all_gpus_grp" group id: 1
```

Set Watches

```
dcgmi health -g 1 --set pmi
Health monitor systems set successfully
```

Get Watches

```
dcgmi health -g 1 -f
+---------+------------------------------------------------------------------+
| Group Health Watches                                                       |
+=========+==================================================================+
| PCIe    |  On                                                              |
| NVLINK  |  Off                                                             |
| PMU     |  Off                                                             |
| MCU     |  Off                                                             |
| Memory  |  On                                                              |
| SM      |  Off                                                             |
| InfoROM |  On                                                              |
| Thermal |  Off                                                             |
| Power   |  Off                                                             |
| Driver  |  Off                                                             |
+---------+------------------------------------------------------------------+
```

Active Health Monitoring & Analysis

NON INVASIVE

Performed during job execution

Overall health for the GPU subsystems (PCIe, SM, MCU, PMU, Inforom, Power and thermal system)

Run Health Check : Healthy System

```
dcgmi health --check -g 1

Health Monitor Report
+------------------+----------------------------------------------------------+
| Overall Health:   Healthy                                                   |
+==================+==========================================================+
```

Run Health Check : System with problems

```
dcgmi health --check -g 1
Health Monitor Report
+-------------------+----------------------------------------------------------+
| Group 1           | Overall Health: Warning                                  |
+===================+==========================================================+
| GPU ID: 0         | Warning                                                  |
|                   | PCIe system: Warning - Detected more than 8 PCIe         |
|                   | replays per minute for GPU 0: 13                         |
+-------------------+----------------------------------------------------------+
| GPU ID: 1         | Warning                                                  |
|                   | InfoROM system: Warning - A corrupt InfoROM has been     |
|                   | detected in GPU 1.                                       |
+-------------------+----------------------------------------------------------+
```

Comprehensive Diagnostics

INVASIVE

Performed at job epilogue/prologue or when job fails

Validates device sub-components, interlink bandwidth, memory/ecc state and deployment software integrity

Several levels of diagnostic are available. Level 1-3 with –r.

Quick Diagnostics (~secs)

```
dcgmi diag -g 1 -r 1
+---------------------------+-------------+
| Diagnostic                | Result      |
+===========================+=============+
|-----  Deployment  --------+-------------|
| Blacklist                 | Pass        |
| NVML Library              | Pass        |
| CUDA Main Library         | Pass        |
| CUDA Toolkit Libraries    | Pass        |
| Permissions and OS Blocks | Pass        |
| Persistence Mode          | Pass        |
| Environment Variables     | Pass        |
| Page Retirement           | Pass        |
| Graphics Processes        | Pass        |
+---------------------------+-------------+
```

Comprehensive Diagnostics

INVASIVE

Performed at job epilogue/prologue or when job fails

Validates device sub-components, interlink bandwidth, memory/ecc state and deployment software integrity

Several levels of diagnostic are available. Level 1-3 with -r.

Extended Diagnostics (~mins)

```
dcgmi diag -g 1 -r 2
+---------------------------+-------------+
| Diagnostic                | Result      |
+===========================+=============+
|-----  Deployment  --------+-------------|
| Blacklist                 | Pass        |
| NVML Library              | Pass        |
| CUDA Main Library         | Pass        |
| CUDA Toolkit Libraries    | Pass        |
| Permissions and OS Blocks | Pass        |
| Persistence Mode          | Pass        |
| Environment Variables     | Pass        |
| Page Retirement           | Pass        |
| Graphics Processes        | Pass        |
+-----  Performance  -------+-------------+
| SM Performance            | Pass - All  |
| Targeted Performance      | Pass - All  |
| Targeted Power            | Warn - All  |
+---------------------------+-------------+
```

Comprehensive Diagnostics

INVASIVE

Performed at job epilogue/prologue or when job fails

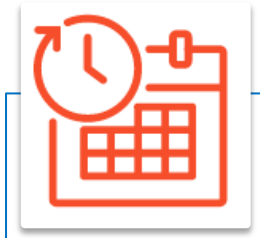
Validates device sub-components, interlink bandwidth, memory/ecc state and deployment software integrity

Several levels of diagnostic are available. Level 1-3 with -r.

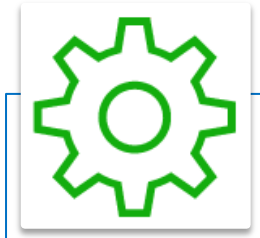
Hardware Diagnostics

```
dcgmi diag -r 3
+---------------------------+------------+
| Diagnostic                | Result     |
+===========================+============+
|-----  Deployment  --------+------------|
| Blacklist                 | Pass       |
| NVML Library              | Pass       |
| CUDA Main Library         | Pass       |
| CUDA Toolkit Libraries    | Pass       |
| Permissions and OS Blocks | Pass       |
| Persistence Mode          | Pass       |
| Environment Variables     | Pass       |
| Page Retirement           | Pass       |
| Graphics Processes        | Pass       |
+-----  Hardware  ----------+------------+
| GPU Memory                | Pass - All |
| Diagnostic                | Pass - All |
+-----  Integration  -------+------------+
| PCIe                      | Pass - All |
+-----  Performance  -------+------------+
| SM Performance            | Pass - All |
| Targeted Performance      | Pass - All |
| Targeted Power            | Warn - All |
+---------------------------+------------+
```

Streamline GPU Administration & TCO



Flexible GPU Governance Policies



Manage GPU group Configuration

Job Statistics

Flexible GPU Governance Policies

With Existing Tools

Continuous monitoring by the user

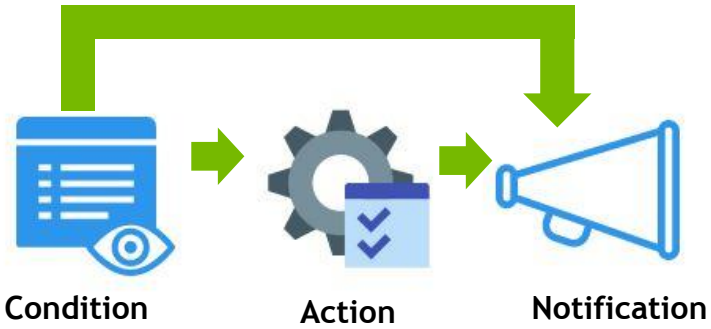
↓

Identify GPUs with double bit errors

↓

Perform GPU reset to correct problems

Using DCGM



Auto-detects double bit errors, performs page retirement, and notifies the user

Condition: Watch for DBE
Action: Page retirement
Notification: Callback

Manage GPU group Configuration

MAINTAINS CONFIGURATION

Initialization: Configure all GPUs (global group)

Per-job basis: Individual partitioned group settings

Maintains settings across driver restarts, GPU resets or at job start

Supports SET, GET and ENFORCE

Get Config for the group of GPUs

```
dcgmi config -g 1 --get
+---------------------------+------------------------+------------------------+
| all_gpu_group             |                        |                        |
| Group of 2 GPUs           | TARGET CONFIGURATION   | CURRENT CONFIGURATION  |
+===========================+========================+========================+
| Sync Boost                |  Disabled              |  Disabled              |
| SM Application Clock      |  705                   |  705                   |
| Memory Application Clock  |  2600                  |  2600                  |
| ECC Mode                  |  Enabled               |  Enabled               |
| Power Limit               |  225                   |  225                   |
| Compute Mode              |  E. Process            |  E. Process            |
+---------------------------+------------------------+------------------------+
```

DCGM maintains the target configuration across resets

Manage GPU group Configuration

MAINTAINS CONFIGURATION

Initialization: Configure all GPUs (global group)

Per-job basis: Individual partitioned group settings

Maintains settings across driver restarts, GPU resets or at job start

Supports SET, GET and ENFORCE

Disable ECC mode [Requires GPU Reset]

```
dcgmi config -g 1 --set -e 0
Configuration successfully set.
```

Get Group config [Note DCGM performed reset]

```
dcgmi config -g 1 --get
+--------------------------+------------------------+------------------------+
| all_gpu_group            |                        |                        |
| Group of 2 GPUs          | TARGET CONFIGURATION   | CURRENT CONFIGURATION  |
+==========================+========================+========================+
| Sync Boost               |  Disabled              |  Disabled              |
| SM Application Clock     |  705                   |  705                   |
| Memory Application Clock |  2600                  |  2600                  |
| ECC Mode                 |  Disabled              |  Disabled              |
| Power Limit              |  225                   |  225                   |
| Compute Mode             |  E. Process            |  E. Process            |
+--------------------------+------------------------+------------------------+
```

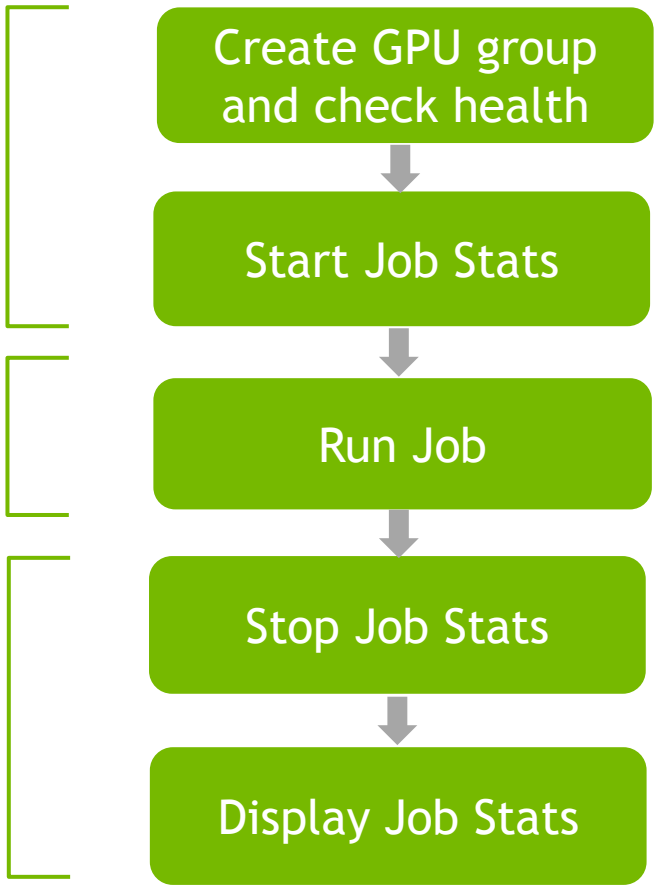
Job Statistics

Which GPUs did my job run on?

How much of the GPUs did my job use?

Any error or warning conditions during my job (ECC errors, clock throttling, etc)

Are the GPUs healthy and ready for the next job?



JOB LIFECYCLE

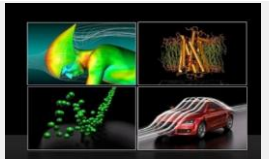
Create GPU group and check health

```
dcgmi group --create demogroup --default
Successfully created group "demogroup"

dcgmi health --check -g 2
Health Monitor Report
+------------------+----------------------+
| Overall Health:   Healthy
|
+==========================================+
```

Start Job Stats

```
dcgmi stats --jstart demojob -g 2
Successfully started recording stats for demojob.
```

Run Job



Stop Job Stats

```
dcgmi stats –jstop demojob -g 2
Successfully started recording stats for demojob.
```

Display Job Stats

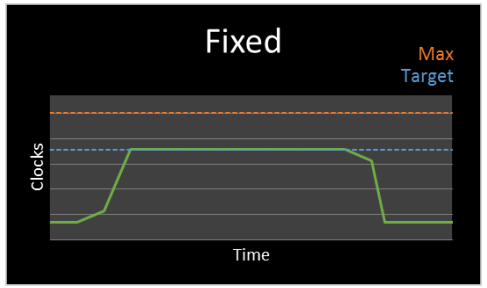
JOB LIFECYCLE

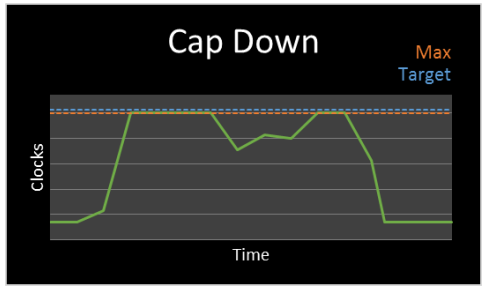
```
dcgmi stats --job demojob -v -g 2
Successfully retrieved statistics for job: demojob.
+-------------------------------------------------------------------------+
| GPU ID: 0                                                               |
+=================================+=======================================+
|-----  Execution Stats  ----------+---------------------------------------|
| Start Time                      | Wed Mar  9 15:07:34 2016              |
| End Time                        | Wed Mar  9 15:08:00 2016              |
| Total Execution Time (sec)      | 25.48                                 |
| No. of Processes                | 1                                     |
| Compute PID                     | 23112                                 |
+-----  Performance Stats  --------+---------------------------------------+
| Energy Consumed (Joules)        | 1437                                  |
| Max GPU Memory Used (bytes)     | 120324096                             |
| SM Clock (MHz)                  | Avg: 998, Max: 1177, Min: 405         |
| Memory Clock (MHz)              | Avg: 2068, Max: 2505, Min: 324        |
| SM Utilization (%)              | Avg: 76, Max: 100, Min: 0             |
| Memory Utilization (%)          | Avg: 0, Max: 1, Min: 0                |
| PCIe Rx Bandwidth (megabytes)   | Avg: 0, Max: 0, Min: 0                |
| PCIe Tx Bandwidth (megabytes)   | Avg: 0, Max: 0, Min: 0                |
+-----  Event Stats  --------------+---------------------------------------+
| Single Bit ECC Errors           | 0                                     |
| Double Bit ECC Errors           | 0                                     |
| PCIe Replay Warnings            | 0                                     |
| Critical XID Errors             | 0                                     |
+-----  Slowdown Stats  ----------+---------------------------------------+
| Due to - Power (%)              | 0                                     |
|        - Thermal (%)            | Not Supported                         |
|        - Reliability (%)        | Not Supported                         |
|        - Board Limit (%)        | Not Supported                         |
|        - Low Utilization (%)    | Not Supported                         |
|        - Sync Boost (%)         | 0                                     |
+---------------------------------+---------------------------------------+
```

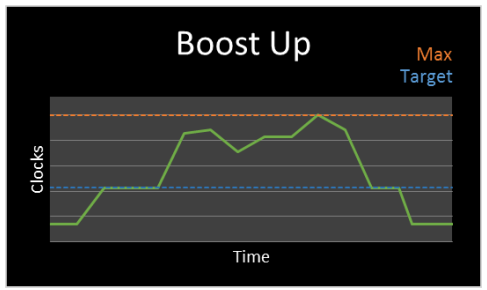
Boost Performance & Resource Efficiency

Enhanced Power & Clock Mgmt.

Enhanced Power & Clock Mgmt.







Dynamic Power Capping

- ✓Drive better power density through dynamic power capping
- ✓Apply power capping to a single or a group of GPUs

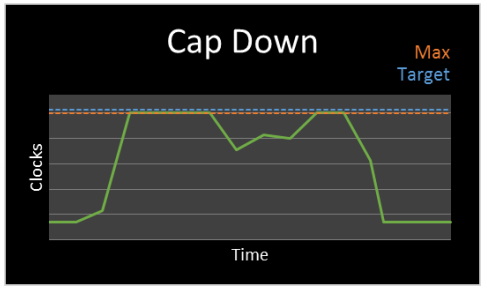
Fixed Clocks

- ✓Target conservative clock rate for fixed performance
- ✓Useful for profiling

Synchronous Clock Boost

- ✓Predictable performance through group GPU clock boost in lockstep
- ✓Dynamically modulate mutli-gpu clocks across multiple boards in unison based on target workload, power budgets or other criteria

Dynamic Power Capping



dcgmi config --get

```
+--------------------------+------------------------+------------------------+
| DCGM_ALL_SUPPORTED_GPUS  |                        |                        |
| Group of 2 GPUs          | TARGET CONFIGURATION   | CURRENT CONFIGURATION  |
+==========================+========================+========================+
| Sync Boost               |  Not Specified         |  Disabled              |
| SM Application Clock     |  Not Specified         |  1000                  |
| Memory Application Clock |  Not Specified         |  3505                  |
| ECC Mode                 |  Not Specified         |  Not Supported         |
| Power Limit              |  Not Specified         |  250                   |
| Compute Mode             |  Not Specified         |  Unrestricted          |
+--------------------------+------------------------+------------------------+
```

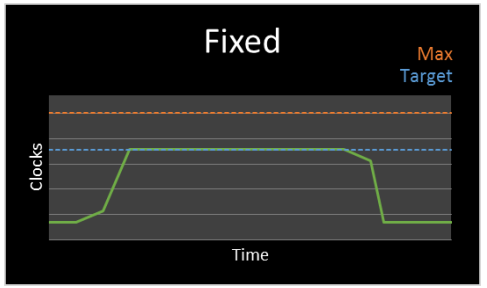
dcgmi config --set -P 200

```
Configuration successfully set.
```

dcgmi config --get

```
+--------------------------+------------------------+------------------------+
| DCGM_ALL_SUPPORTED_GPUS  |                        |                        |
| Group of 2 GPUs          | TARGET CONFIGURATION   | CURRENT CONFIGURATION  |
+==========================+========================+========================+
| Sync Boost               |  Not Specified         |  Disabled              |
| SM Application Clock     |  Not Specified         |  1000                  |
| Memory Application Clock |  Not Specified         |  3505                  |
| ECC Mode                 |  Not Specified         |  Not Supported         |
| Power Limit              |  200                   |  200                   |
| Compute Mode             |  Not Specified         |  Unrestricted          |
+--------------------------+------------------------+------------------------+
```

Fixed Clocks



dcgmi config --get

```
+--------------------------+------------------------+------------------------+
| DCGM_ALL_SUPPORTED_GPUS  |                        |                        |
| Group of 2 GPUs          | TARGET CONFIGURATION   | CURRENT CONFIGURATION  |
+==========================+========================+========================+
| Sync Boost               |  Not Specified         |  Disabled              |
| SM Application Clock     |  Not Specified         |  1000                  |
| Memory Application Clock |  Not Specified         |  3505                  |
| ECC Mode                 |  Not Specified         |  Not Supported         |
| Power Limit              |  Not Specified         |  250                   |
| Compute Mode             |  Not Specified         |  Unrestricted          |
+--------------------------+------------------------+------------------------+
```

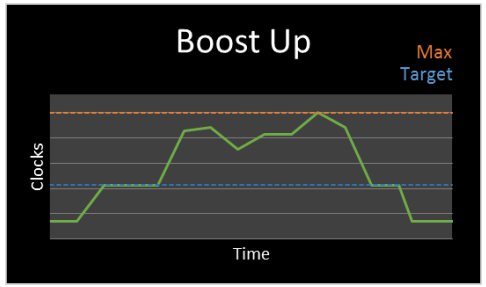
dcgmi config --set -a 3505,1215

```
Configuration successfully set.
```

dcgmi config --get

```
+--------------------------+------------------------+------------------------+
| DCGM_ALL_SUPPORTED_GPUS  |                        |                        |
| Group of 2 GPUs          | TARGET CONFIGURATION   | CURRENT CONFIGURATION  |
+==========================+========================+========================+
| Sync Boost               |  Not Specified         |  Disabled              |
| SM Application Clock     |  1215                  |  1215                  |
| Memory Application Clock |  3505                  |  3505                  |
| ECC Mode                 |  Not Specified         |  Not Supported         |
| Power Limit              |  Not Specified         |  250                   |
| Compute Mode             |  Not Specified         |  Unrestricted          |
+--------------------------+------------------------+------------------------+
```

Synchronized Boost Clocks



dcgmi config --get

```
+--------------------------+------------------------+------------------------+
| DCGM_ALL_SUPPORTED_GPUS  |                        |                        |
| Group of 2 GPUs          | TARGET CONFIGURATION   | CURRENT CONFIGURATION  |
+==========================+========================+========================+
| Sync Boost               |  Not Specified         |  Disabled              |
| SM Application Clock     |  Not Specified         |  1000                  |
| Memory Application Clock |  Not Specified         |  3505                  |
| ECC Mode                 |  Not Specified         |  Not Supported         |
| Power Limit              |  Not Specified         |  250                   |
| Compute Mode             |  Not Specified         |  Unrestricted          |
+--------------------------+------------------------+------------------------+
```

dcgmi config --set -s 1

```
Configuration successfully set.
```

dcgmi config --get

```
+--------------------------+------------------------+------------------------+
| DCGM_ALL_SUPPORTED_GPUS  |                        |                        |
| Group of 2 GPUs          | TARGET CONFIGURATION   | CURRENT CONFIGURATION  |
+==========================+========================+========================+
| Sync Boost               |  Enabled               |  Enabled               |
| SM Application Clock     |  Not Specified         |  1000                  |
| Memory Application Clock |  Not Specified         |  3505                  |
| ECC Mode                 |  Not Specified         |  Not Supported         |
| Power Limit              |  Not Specified         |  250                   |
| Compute Mode             |  Not Specified         |  Unrestricted          |
+--------------------------+------------------------+------------------------+
```

HOW SHOULD I MANAGE MY GPUS?

NVML

Stateless queries. Can only query current data

Low overhead while running, high overhead to develop

Management app must run on same box as GPUs

Low-level control of GPUs

DCGM

Can query a few hours of metrics

Provides health checks and diagnostics

Can batch queries/operations to groups of GPUs

Can be remote or local

3RD PARTY TOOLS

Provide database, graphs, and a nice UI

Need management node(s)

Development already done. You just have to configure the tools.

WHICH GPUS ARE SUPPORTED

Tesla GPUs K80 and Newer

Tesla-recommended Driver r361 or later (Includes hardware diagnostic!)

Requires an additional DCGM package

WAKE-UP MODES

TIMED

Wake up when work is due.

Provides consistent, fixed-interval samples

Use when you don't mind DCGM using a small, recurring amount of CPU.

Can automatically enforce policy

LOCK-STEP

DCGM only wakes up when called.

Samples only taken when requested.

No jitter. DCGM asleep unless requested to wake up.

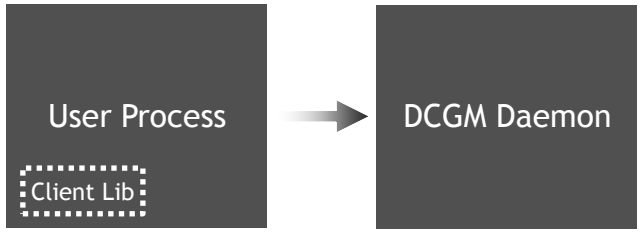
DCGM MODES OF OPERATION

STANDALONE

Runs as daemon

Client libraries connect via TCP/IP

1 DCGM for several clients



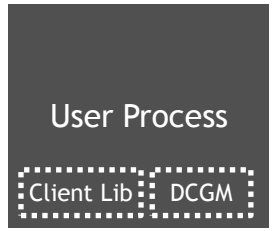
EMBEDDED

Runs within client process

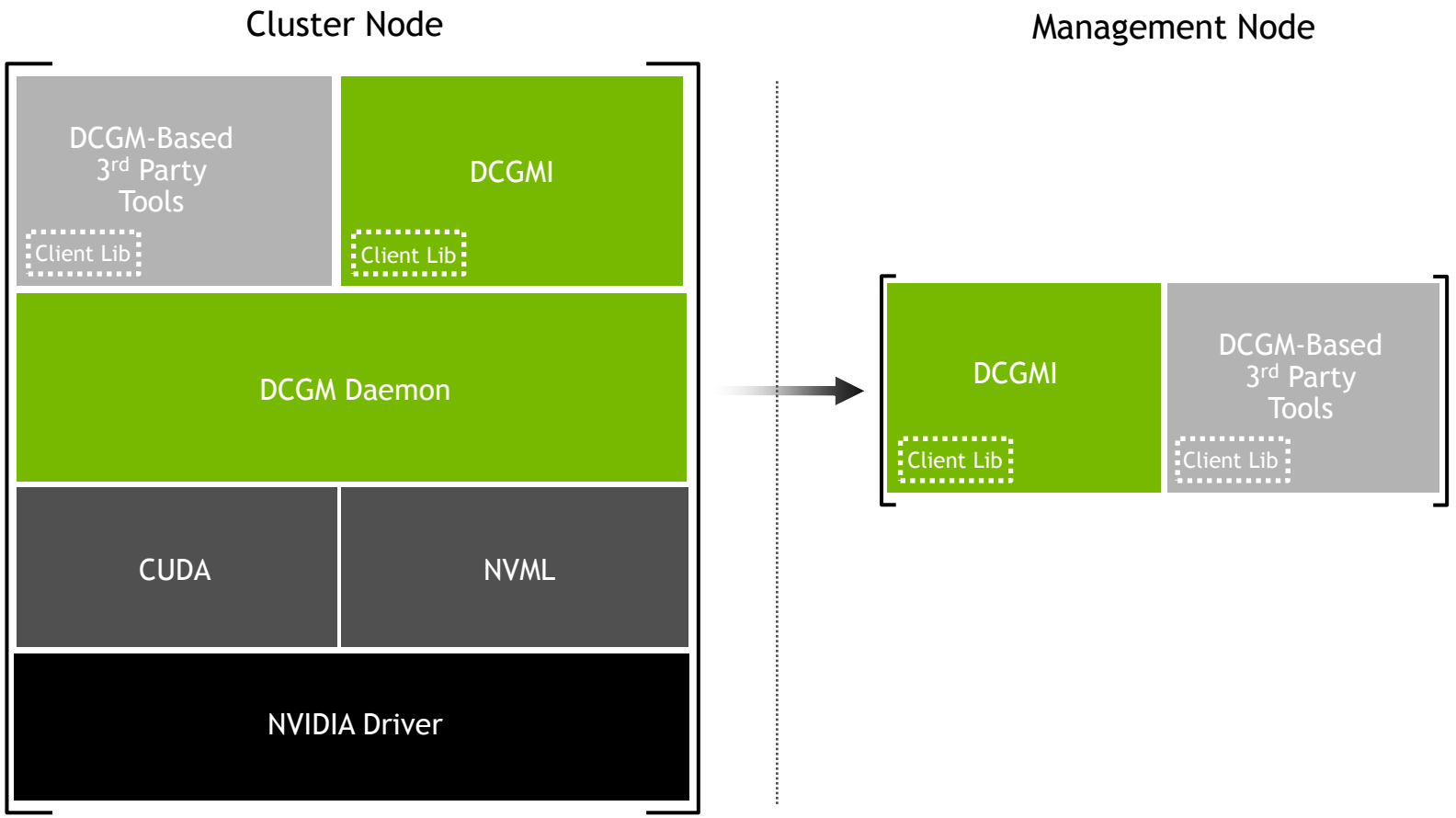
Even within python

1 DCGM per client process

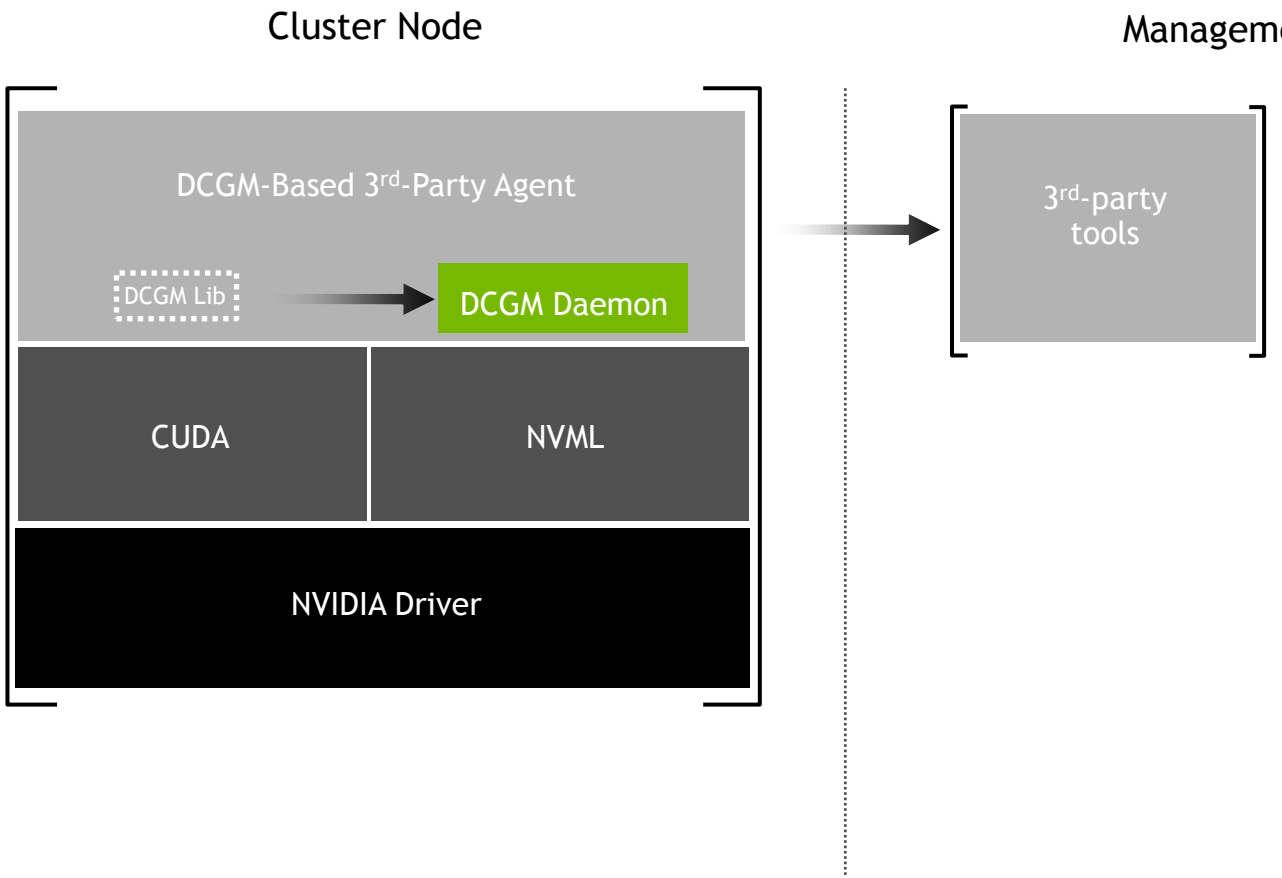
No TCP/IP necessary



WHERE DCGM FITS IN - STANDALONE



WHERE DCGM FITS IN - EMBEDDED



WHERE IS DCGM INSTALLED?

DCGM SDK Headers	*/usr/include*
DCGM Libraries	*/usr/lib*
C and python samples	*/usr/src/dcgm/sdk_samples*
Python bindings	*/usr/src/dcgm/bindings*
DCGMI and nv-hostengine	*/usr/bin*
User guide and License	*/usr/share/doc/datacenter-gpu-manager*

PYTHON BINDINGS

First-class, not just C-style

Object-oriented

Documented independently. No more referring to C APIs

Designed with usability in mind

C bindings are still first-class as well

PYTHON BINDINGS - SAMPLE

```
#Old C-Style
def callback(gpuId, values, numValues, userData):
      values[gpuId][values[0].fieldId] = values[0:numValues]
      return 0

handle = dcgmInit(host, DCGM_OPERATION_MODE_AUTO)
groupId = dcgmGroupCreate(handle, DCGM_GROUP_DEFAULT, “mygroup”)
dcgmWatchFields(handle, groupId, CLOCKS, 1000000, 3600.0, 0)
values = {}
dcgmGetLatestValues(handle, groupId, CLOCKS, callback, values)


#New and improved style
handle = DcgmHandle(None, host, DCGM_OPERATION_MODE_AUTO)
dcgmGroup = handle.GetSystem().GetDefaultGroup()
dcgmGroup.samples.WatchFields(CLOCKS, 1000000, 3600.0, 0)
values = dcgmGroup.samples.GetLatest(CLOCKS)
```

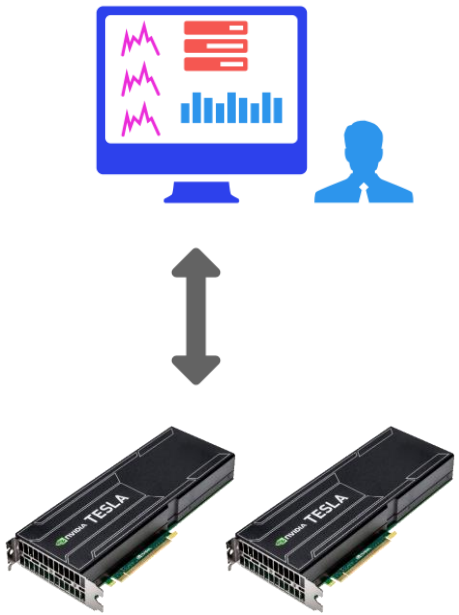
C-Style callback

Values simply returned

C BINDINGS - SAMPLE

```
//Connect to DCGM
result = dcgmInit(ipAddress, DCGM_OPERATION_MODE_AUTO, &dcgmHandle);

//Create a group of GPUs containing all GPUs on the system
result = dcgmGroupCreate(dcgmHandle, DCGM_GROUP_DEFAULT, "test_group", &myGroupId);

//Watch health fields for our group
healthSystems = (dcgmHealthSystems_t) (DCGM_HEALTH_WATCH_PCIE | DCGM_HEALTH_WATCH_MEM);
result = dcgmHealthSet(dcgmHandle, myGroupId, healthSystems);

//Wait for the health fields to update
dcgmUpdateAllFields(dcgmHandle, 1);

//Fetch the health of all GPUs
result = dcgmHealthCheck(dcgmHandle, myGroupId, &results);

//Check the group's overall health
if (results.overallHealth == DCGM_HEALTH_RESULT_PASS)
   printf("Group is healthy!\n");
else
{
   printf("Group is unhealthy\n");
   //TODO: Look at each results.gpu[i] to see which GPUs are unhealthy
}
```

PUBLISHING METRICS EXTERNALLY



DCGM meant to cache 1-4 hours of data

Hope to publish metric-pushing plugins for various TSDBs in the future

Planning to contribute open source plugins for popular metric publishing and TSDB products.

DCGM METRICS IN GRAFANA

DCGM IN UPCOMING PRODUCT RELEASES

GPU TECHNOLOGY CONFERENCE

April 4-7, 2016 | Silicon Valley

THANK YOU

JOIN THE CONVERSATION

#GTC16

JOIN THE NVIDIA DEVELOPER PROGRAM AT developer.nvidia.com/join

JOIN OUR DATA CENTER MANAGEMENT HANGOUT IN POD A FROM 14:00 – 15:00

PRESENTED BY

NVIDIA

GROUP MANAGEMENT

All DCGM operations on GPU groups

Create/Destroy/Modify collection of GPUs on local node

Collection of GPUs as a single abstract resource (correlated to scheduler's notion of node level job)

Global groups (all GPUs in the system): Useful for node level concepts such as global configuration/health

Partitioned groups (subset of GPUs) : Useful for job-level concepts such as job stats and health

METRIC GROUPS

WHY?

Called Field Collections in DCGM

Less Code For Users

Logical grouping of fields

METRIC GROUP EXAMPLES

GPU METADATA

- Brand
- UUID
- VBIOS Version
- PCI Bus ID
- Product Name

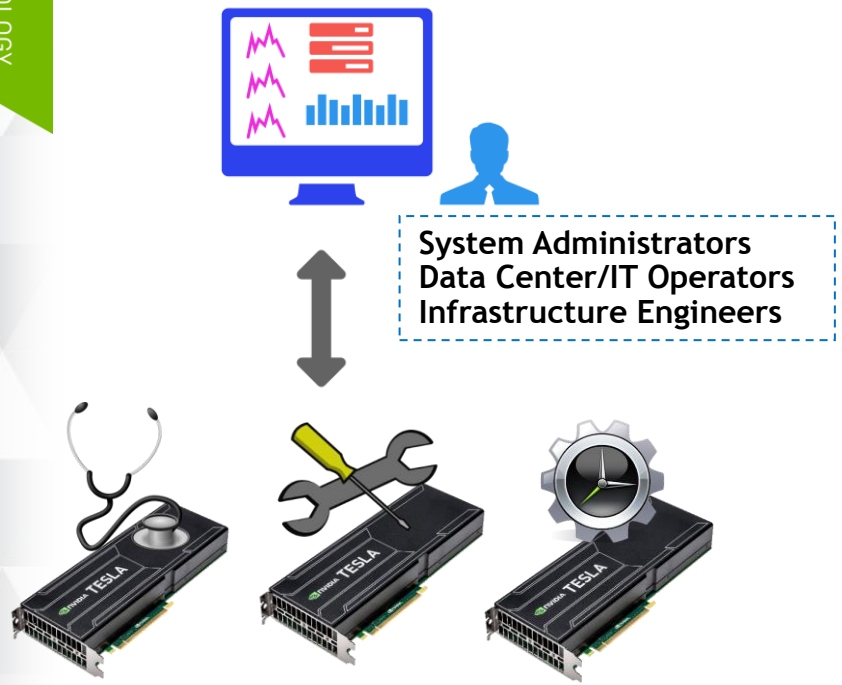
VIOL COUNTERS

- Power Violations
- Thermal Violations
- Voltage Limit
- Low Utilization
- Sync Boost

CLOCKS

- Current Clocks
- Application Clocks
- Clock Samples

DEVICE MGMT. TOOLS - AVAILABLE TODAY



Device & Clock Management

GPU-Aware Job Scheduling

GPU Health Monitoring

Device-level GPU Monitoring Tools – Available Today
NV Mgmt. Library (NVML)
NVIDIA-SMI (Command Line Tool)
Health Monitoring Tool (HealthMon)