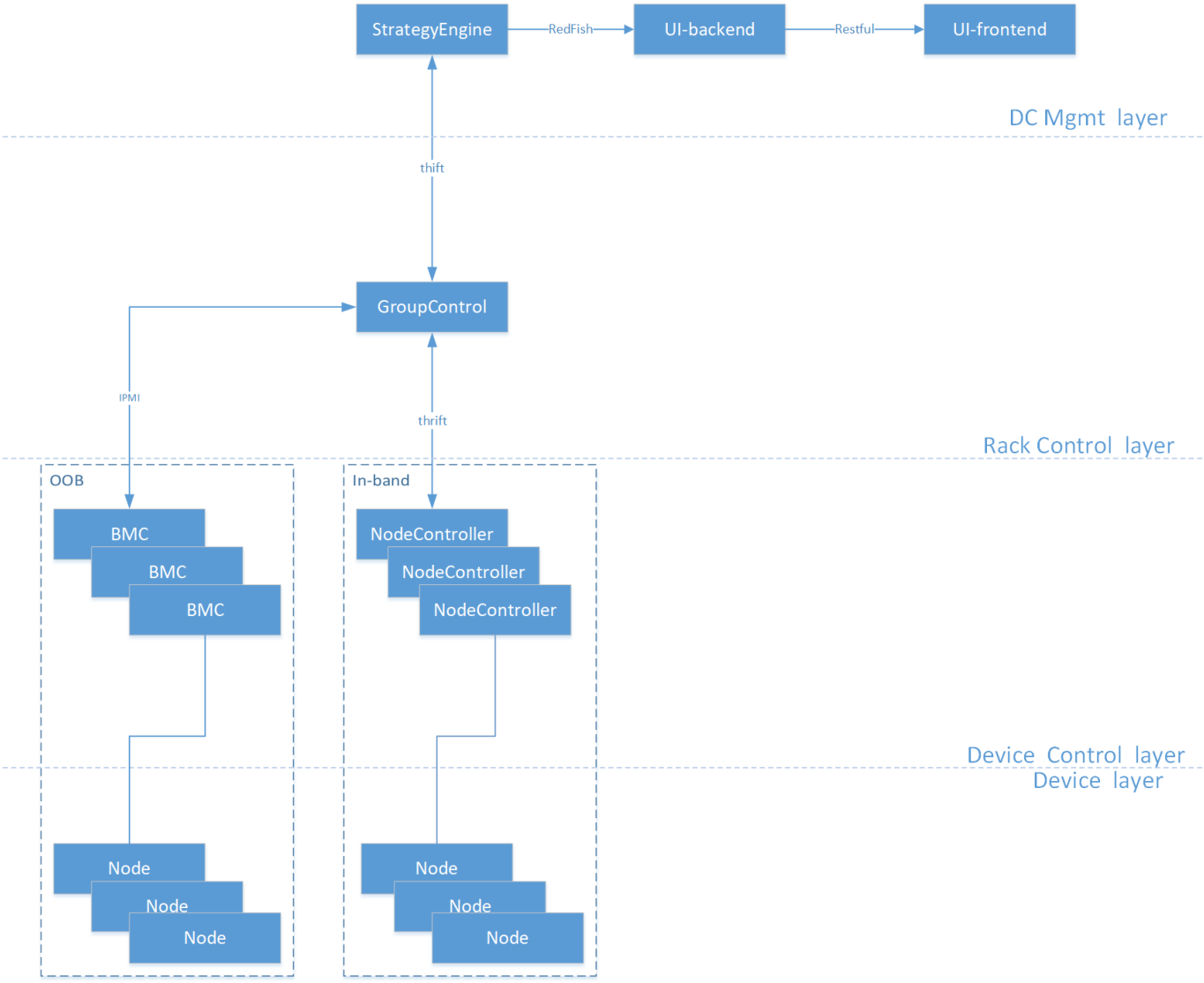
TurboRack - GPC

Group Power Capping (GPC) is a sub-project under TurboRack. This document will describe the architecture and installation instructions of Group Power Capping(GPC). GPC uses Running Average Power Limit(RAPL) and Advanced Rack Level Power management algorithms to optimize rack level power utilization and rack density. If you want to use Smart BBS to get more rack density please refer to the TurboRack - SmartBBS document.

Index

- TurboRack - GPC
  - Index
  - Overview
  - Getting Started
    - System Requirements
    - Setup Group Controller
    - Setup Node Agent(in band node only)
    - Quick node deploy
      - deploy with binary package
    - Workload
  - Architecture
    - Power Capping Policy
  - Services
    - Controller
    - Node Agent
    - Ports
    - Config Folder
    - LOG Folder

Overview



The key features of Snap are:

Architecture: TurboRack has several services

- **Node Controller** - Node Controller is an agent deploy in the server/node side. It can uses for read the power and control the power. It connects to the Group Controller for rack level control. For Out of band power control, this service is not required.
- **Group Controller** - Group Controller connects to the nodes in the rack. and expose the telemetry and control api to strategy engine.
- **Strategy Engine** - Strategy engine gets data from group controller and do some analysis and send action to BBS or node via group controller and node controller.
- ~~**Internal API Service** - to address the Group Controller as an Restful API. easy to integrate to 3rd-part dc control service.~~

Getting Started

this is a reference version, please keep the most of the configurations to default, if it's not mentioned in this document

System Requirements

Turborack - GPC Requires:

- **Centos7.2**
- **Python2.7**
- **Thrift**
- **IPMI Tool (node)**
- **Nginx(Web)**

Setup Node Agent(in-band mode only)

1. Install node service

```
./turborack-install.bin -i node
```

2. Configure node service

```
vim /etc/turbo-rack/node_config.json
{
    "Id": "node_1",
    "PowerMax": 400,
    "PowerMin": 200,
    "PowerRead": {
        "Interface": "Sensor",
        "Method": "IPMI",
        "DCEstimated": false,
        "Sensors": [
            "PS1 Input Power",
            "PS2 Input Power"
        ]
    },
    "PowerControl": {
        "Method": "IMPI"
    }
}
```

- Configure Power Read method and interface:
  Available values for PowerRead.Method: IPMI
  Available values for PowerRead.Interface: Sensor, Raw
  Option 1: read power by sensor mode:
  set PowerRead.Interface = Sensor
  use `ipmitool sensor list` to get the Sensor name of Power Supplies
  e.g. `PS1 Input Power`, `PS2 Input Power`
  Option 2: read power by Raw command
  set PowerRead.Interface = Raw

- Configure Power Control Method:
  Available values for PowerControl.Method: IPMI, RAPL
  Option 1: control power by NodeManager
  set PowerControl.Method = IPMI Option 2: control power by RAPL msr
  set PowerControl.Method = RAPLs

NOTE: PowerRead.Sensors and PowerRead.DCEstimated option only take effective when PowerRead.Interface is set to Sensor

3. Run(restart) node service

```
service turborack-node restart
```

4. Check node service '''shell tail -f /var/log/turbo-rack/node.log ''' The node service is normally run if the value of actual_power is greater than zero.
   - -DEBUG {'uuid': u'bbu-demo-10', 'capping_power': '296', 'min_power': '200', 'max_power': '400', 'time': '1553186500.72', 'actual_power': '266'}

Setup Group Controller

1. Install controller

```
./turborack-install.bin –i controller
```

2. Configure Group Service

   Add the nodes to the node list, set the power capacity of this group

```
vim /etc/turbo-rack/group_config.json

{
    "Id": "group_1",
    "NodeList": [
        {
            "Id": "node_1",
            "Ip": "10.0.1.101",
            "BMCIP": "10.0.2.101",
            "BMCPORT": 623,
            "BMCUSER": "root",
            "BMCPASSWORD": "superuser",
            "Method": "OutOfBand"
        },
        {
            "Id": "node_2",
            "Ip": "10.0.1.102",
            "Method": "InBand"
```

```
        }
    ],
    "PowerCapacity": 2000
}
```

- add node to the list, and config to `InBand` or `OutOfBand` mode.
  option 1: InBand
  controller communicate with the node by node agent
  option 2: OutOfBand
  controller communicate with the node by BMC through ipmi overlan

  NOTE: for nodes with In Band method, do not need to config BMC Related items

3. Run(restart) group service

```
service turborack-group restart
```

4. Configure Strategy Service

```
vim /etc/turbo-rack/group_strategy.json
{
    "power_capping_enable": true,
    "smart_bbu_enable": false,
    "bbu_discharge_info_enable": false,
    "bbu_discharging_percentage": 85,
    "bbu_id": "bbu_1",
    "bbu_ip": "127.0.0.1",
    "bbu_port": 11002,
    "group_id": "group_1",
    "group_ip": "127.0.0.1",
    "group_port": 11000,
    "policy": "maximum",
    "port": 8011,
    "power_capping_effective_percentage": 97,
    "power_provision": 2000
}
```

5. Run(restart) strategy service restart strategy service

```
service turborack-strategy restart
```

6. Check controller service '''shell tail -f /var/log/turbo-rack/group.log '''

- The controller service is normally run if the logs are as follows and there are no error information.
  -INFO {'node_uuid': u'tsi_en_3', 'node_ip': u'xxx.xxx.xxx.xxx', 'capping_power': 240, 'max_power': 400,

'min_power': 200, 'actual_power': 210, 'allocated_power': 0, 'op': 'InBand'}

'''shell tail -f /var/log/turbo-rack/strategy.log '''

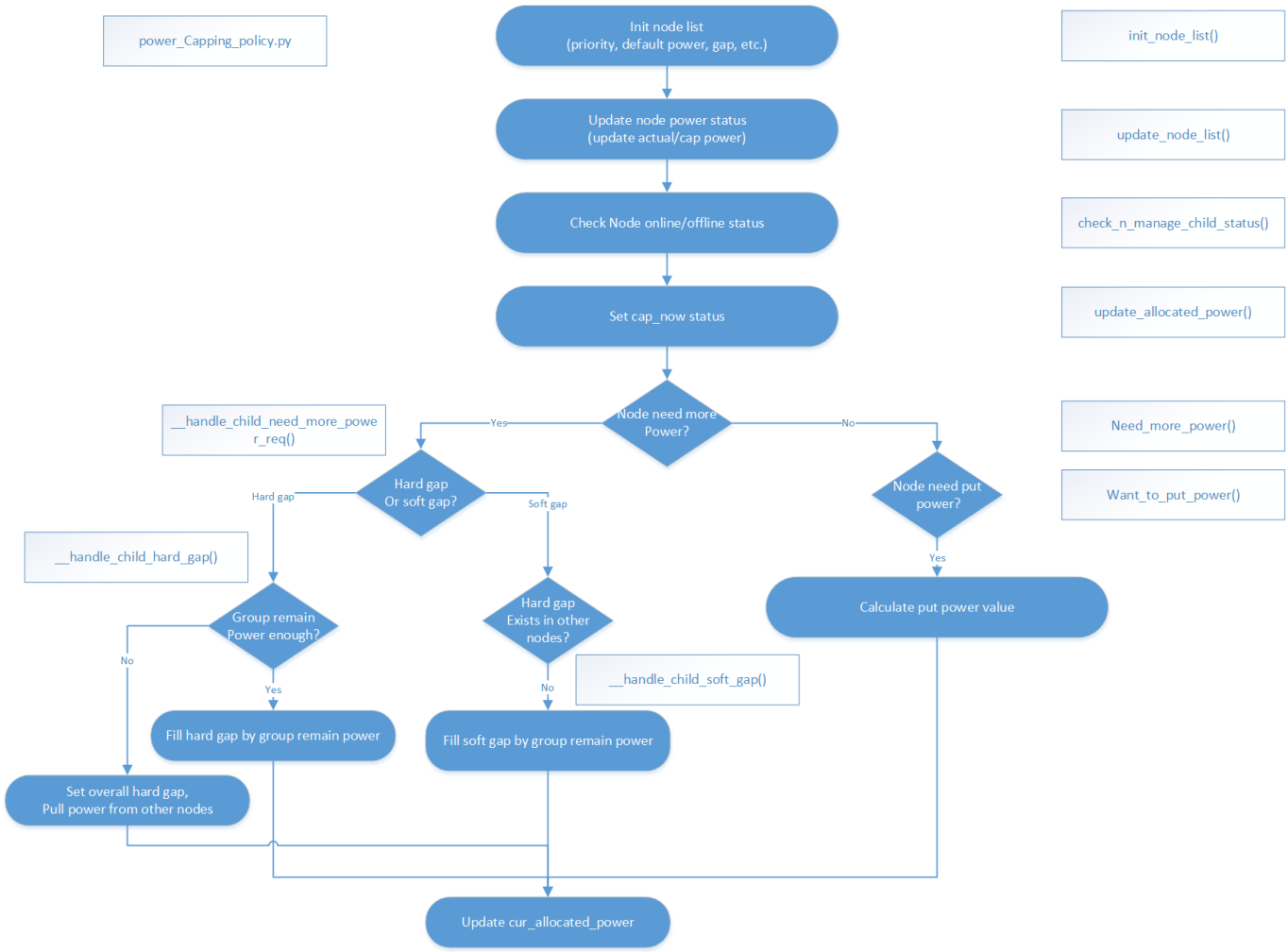
- The controller service is normally run if the logs are as follows and there are no error information.
  -INFO id:bbu-demo-09,capping power:296,actual power:266,allocated power:296, cap_now: True

Workload

We build a workload generation and experiment tool WorkloadGenerator You can use this tool to do some power capping. This tool was integrated to the turborack as well. You can run the demo scenario in GUI.

Architecutre

Power Capping Policy



Services

Controller

- turborack-group (old name GroupController)
- ~~turborack-api (old name Rest API)~~
- turborack-strategy (old name GroupCore)

Node Agent

turborack-node (old name NodeController)

Ports

Role	Name	Port	Description
Controller	turborack-group	11000	thrift
Controller	turborack-strategy	8010	Restful
Node	turborack-node	11001	thrift
Node	wlgen-agent	12000	rpc
Web	UI Backend	50001	Restful
Web	UI Frontend	9000/80	HTTP

(the details of workload refers WorkloadGenerator)

Config Folder

/etc/turbo-rack

LOG Folder

/var/log/turbo-rack