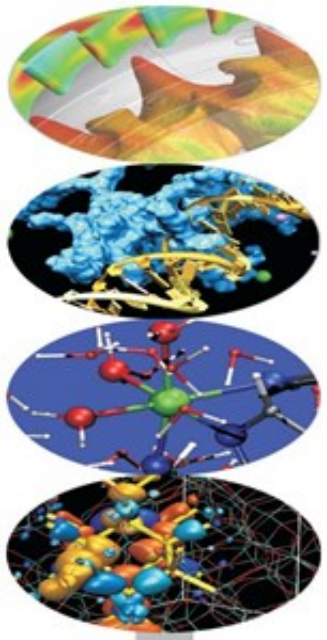
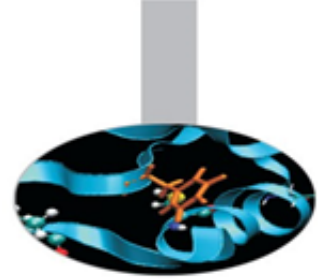
CINECA SCAI
SuperComputing Applications and Innovation



hpcmd tool

HPC User Support @ CINECA
April 14th, 2021

CINECA

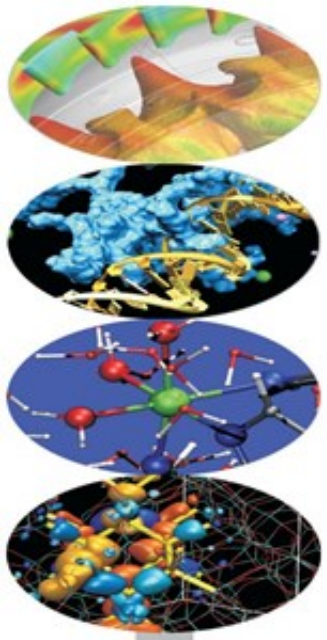


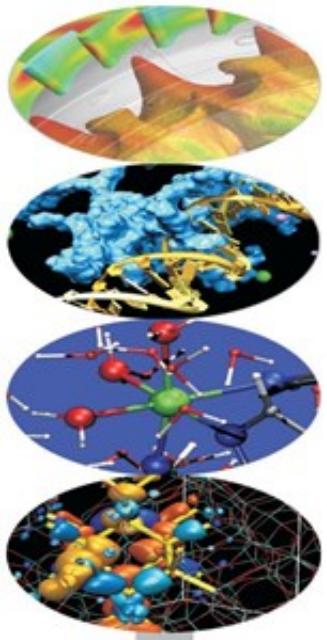
Content

I. Description of the tool

II. Overview of preliminary systemd mode configuration on Marconi cluster

1. Metrics configuration
2. Data transport & collection configuration
3. Data visualization

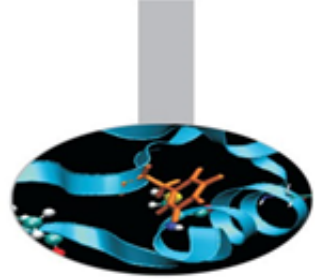




I. Description of the tool

I. hpcmd: description of the tool

http://mpcdf.pages.mpcdf.de/hpcmd/index.html



hpcmd is software daemon that runs Linux perf and comparable tools periodically to obtain metrics from performance counters.

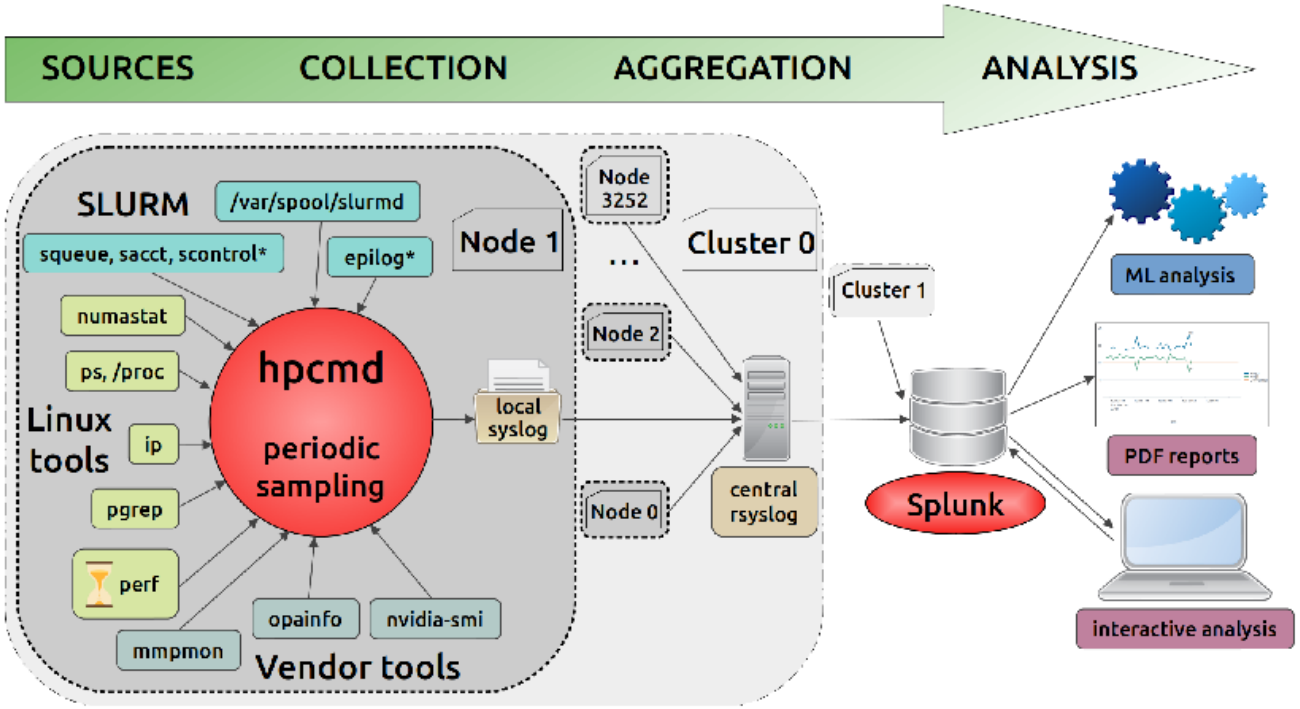
Intel Broadwell, Skylake, and newer processors are fully supported, e.g., to compute the performance in GFLOPS or to obtain the memory bandwidth in GB/s.

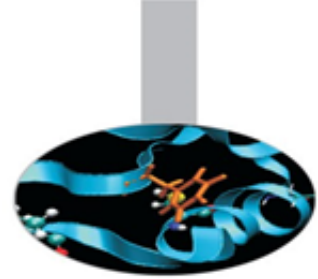
Moreover, performance metrics from GPUs, OmniPath and InfiniBand networks, and GPFS file systems are supported.

hpcmd computes derived metrics and writes the data to syslog lines.

On a cluster installation, these local syslog lines can be collected via rsyslog and finally stored and analyzed in Splunk.

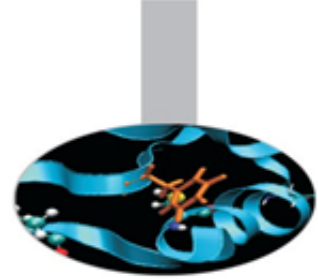
hpcmd fully integrates with the SLURM batch system, enabling to correlate performance metrics with each job.





Top features

- non-measurable performance impact on the applications
- Linux daemon and systemd service
- user mode for custom measurements
- user-triggered suspension of the systemd service
- SLURM integration, SLURM job detection
- flexible hierarchical config files in YAML format



Installation

Basic installation (user mode)

```
$ python setup.py install --user
```

Develop installation (to test-drive the package)

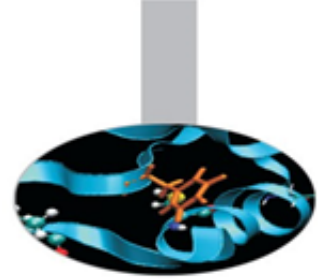
```
$ python setup-py develop --user
```

1. `# echo 0 >/proc/sys/kernel/perf_event_paranoid` to allow access to the perf counters (as root)
2. `hpcmd -d` to launch the daemon
3. run `cat /var/log/syslog` to see the live measurement data (or `cat /var/log/messages` or `journalctl`)
4. `hpcmd -k` to shut down the daemon cleanly

Large scale deployment via RPM

An RPM spec file is provided which can be used to package hpcmd as an RPM for the deployment on HPC systems.

Sources & collection

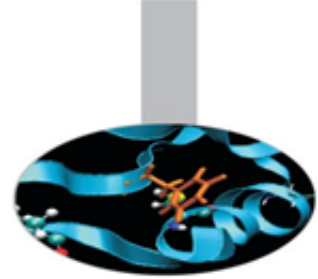


hpcmd is a "software daemon" that periodically executes the following binaries to measure job performance on each compute node:

- perf
- top
- ps
- numastat
- ibstat (InfiniBand)
- opastat (OmniPath)
- ipstat (Ethernet)
- nvidia-smi (GPU)
- mmpmon (GPFS)

It is also integrated with SLURM, it collects information for job running in each node by making periodic checks in the local directory /var/spool/slurmd.

All these metrics can be configured in the configuration file for the specific system: config/<myplatform.yaml>.



Data transport & collection (aggregation)

On each node it generates a log file which is then transported (via ethernet) and collected by a central rsyslog server.
[on a dedicated node?]

Data analysis

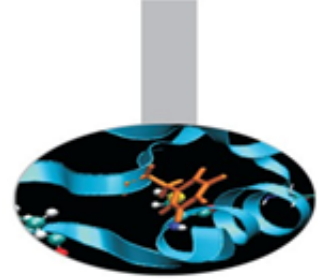
since data generated by the tool on each node is aggregated by the rsyslog, it is therefore possible to convert this data back to other output formats as the required by elasticsearch

Other considerations

storage: the storage required for the retention of job data generated in 1 year is <= 1TB, assuming that the order of magnitude of traffic is also equivalent on Marconi SKL.

database: a more in-depth study of the infrastructure necessary for database management is also required, taking into account the forecast increase in the volume of data stored & to be preserved.

ownership and security of data collected by nodes



=== http://mpcdf.pages.mpcdf.de/hpcmd/reporting.html ===

We are currently logging approximately 1.5 GiB of data per day for two large MPCDF clusters combined.

In order to make Splunk queries faster, this data is stored in two separate databases:

The primary database contains all the measurements for all the jobs, and it takes most of the storage space.

For each job, the secondary database only contains one message from the beginning and the job summary, i.e., the most important metrics to characterize the job.

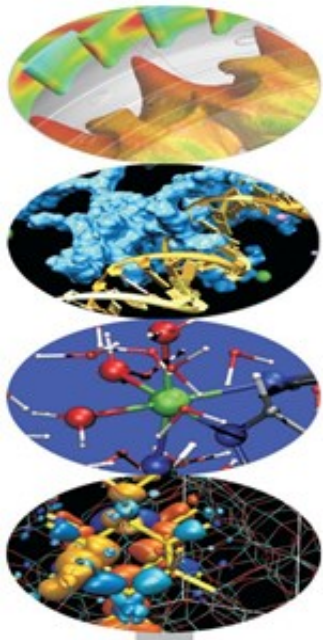
This design was chosen to make it possible to implement a quick overview of all jobs in a roofline view efficiently.

Moreover, it allows SPLUNK to perform faster as any query can start by doing a sub-query on a secondary database which narrows down the main search on the primary database. The resulting hierarchical queries are complex, but they are much more efficient, and thus drastically improve the user experience.

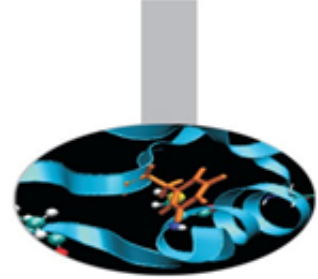
Alternative frameworks to SPLUNK such as ELK or custom solutions can certainly be used for the analysis with similar success.

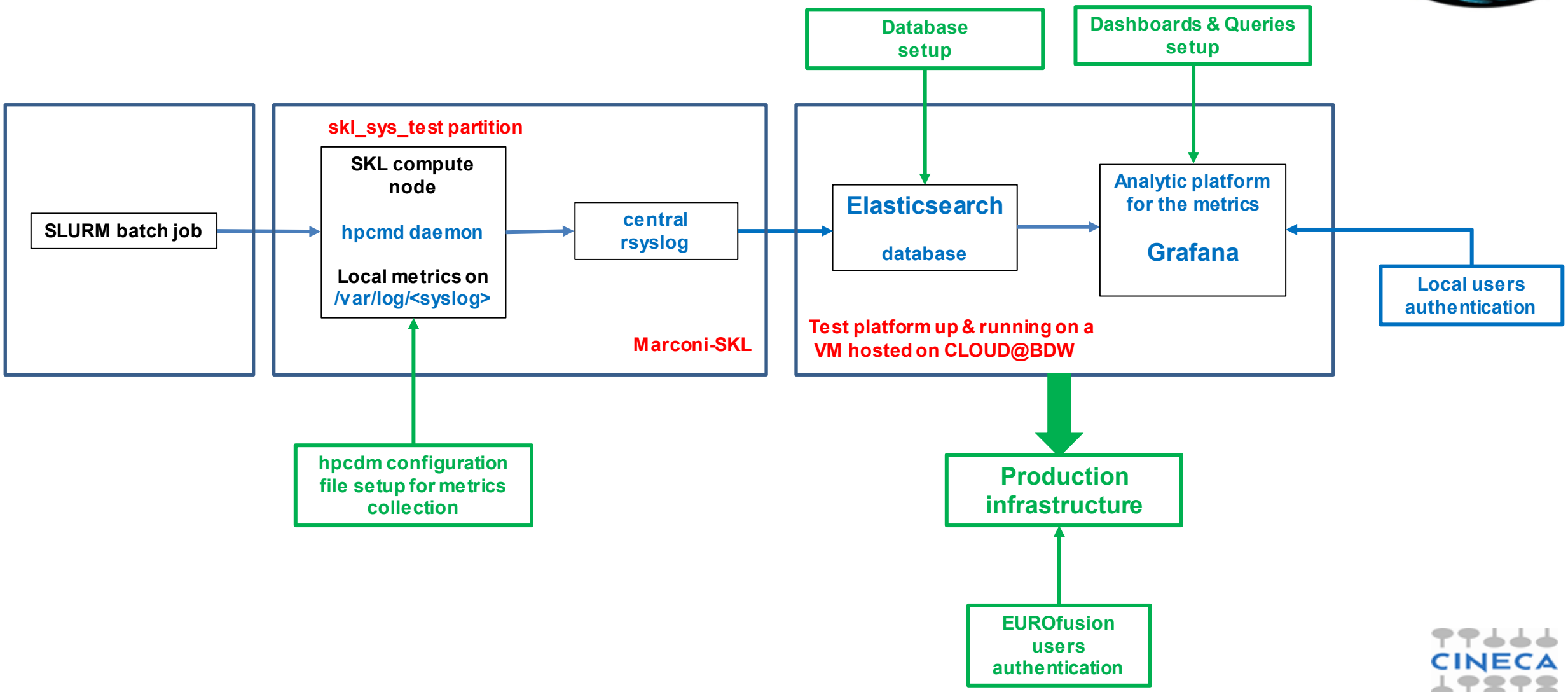
Our main motivation for choosing SPLUNK is that it was already configured and successfully used on MPCDF systems for other purposes, and it provides simple yet powerful tools to perform all the analysis and visualization tasks necessary.

=== === ===

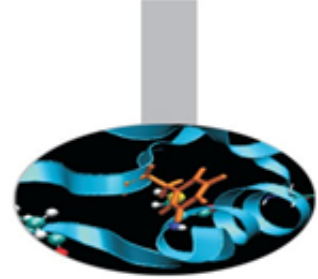


II. Preliminary configuration on systemd mode on Marconi cluster




Database setup
Dashboards & Queries setup
skl_sys_test partition
SKL compute node
hpcmd daemon
Local metrics on /var/log/<syslog>
SLURM batch job
central rsyslog
Elasticsearch
database
Analytic platform for the metrics
Grafana
Local users authentication
Marconi-SKL
Test platform up & running on a VM hosted on CLOUD@BDW
hpcdm configuration file setup for metrics collection
Production infrastructure
EUROfusion users authentication

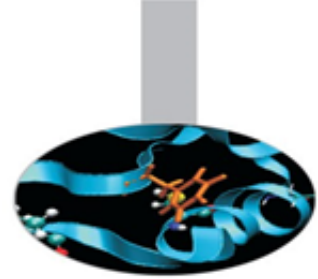
III.1. hpcmd local metrics collection

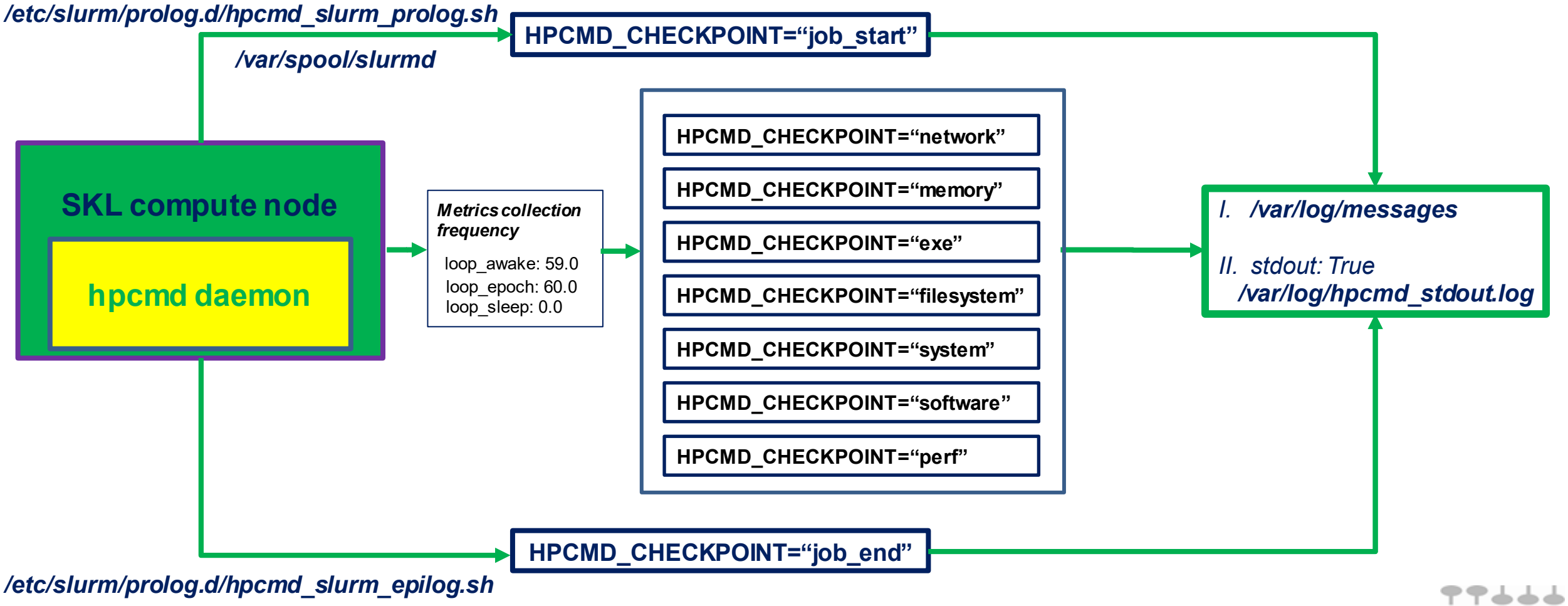


We are typically monitoring only nodes which have a single job running on them, i.e., data is not collected for nodes that are currently idle or shared, as such cases are considered less relevant in our context and would be much harder to interpret.

https://link.springer.com/chapter/10.1007%2F978-3-030-48340-1_47

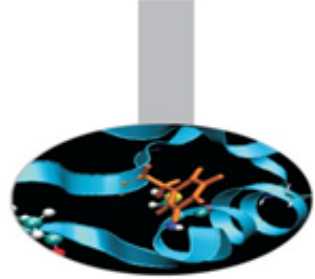
III.1. hpcmd local metrics collection





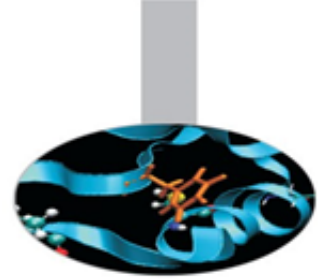
Features status

https://gitlab.mpcdf.mpg.de/mpcdf/hpcmd/-/tree/d3ffdf0419f7cd5cc16e8a459ceb3452aa38c249



feature	Marconi systemd	To Do
PROLOG trigger		---
job_start message		---
EPILOG trigger		---
job_summary msg		Export new path to slurm binaries (sacct, squeue, scontrol) that has changed after SLURM upgrade
suspend and resume		To verify the correct functioning
Systemd service		To install systemd service file for hpcmd: > cp hpcmd.service /etc/systemd/system/hpcmd.service

hpcmd: daemon configuration



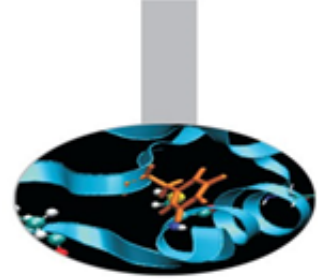
default.yaml

```
daemon:
  suspend: false
  loop_awake: 590.0
  loop_epoch: 0.0
  loop_sleep: 10.0
  pid_file: /tmp/hpcmd.pid
  working_directory: /tmp
  trigger_directory: /tmp/hpcmd

output:
  debug: false
  debug_lvl: DEBUG
  syslog: true
  intro: true
  epilog: true
  stdout: false
  prefix: ''
  out_file: stdout
  err_file: stderr
  epilog_file: /tmp/hpcmd/epilog
  cfg_out: /tmp/hpcmd.yaml
  tagid: HPCMD_CHECKPOINT
  userid: true
  local_slurm_jobs_tag: false
```

marconi_system.yaml

```
daemon:
  loop_awake: 59.0
  loop_epoch: 60.0
  loop_sleep: 0.0
  pid_file: /var/run/hpcmd.pid
  working_directory: /var/lib/hpcmd

output:
  cfg_out: /var/log/hpcmd.yaml
  stdout: true
  out_file: /var/log/hpcmd_stdout.log
  err_file: /var/log/hpcmd_stderr.log
```

hpcmd: slurm configuration



default.yaml

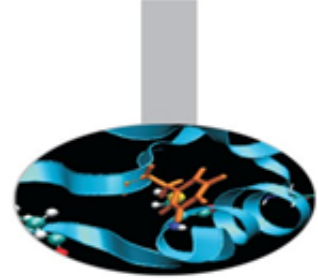
```
slurm:
  <<: *base
  node_blacklist: ‘’
  slurm_conf: /var/spool/slurmd/conf-cache/slurm.conf
```

marconi_system.yaml

```
slurm:
  enable: true
```

hpcmd's collection of SLURM-related functions.

hpcmd: perf configuration



default.yaml

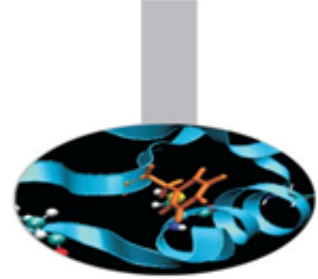
```
perf: # "perf stat -x , -a -e events cfg['perf']['agg-counts'] sleep cfg['daemon']['loop_awake']
  <<: *base
  agg-counts: --per-socket
  postfix: u
  generic:
    <<: *base
    events:
      alignment-faults: false
      br_misp_retired.all_branches: false
      branch-misses: true
      branches: true
      bus-cycles: false
      cache-misses: true
      cache-references: true
      context-switches: false
      cpu-migrations: false
      cycles: true
      emulation-faults: false
      idq.ms_uops: false
      instructions: true
      machine_clears.count: false
      major-faults: true
      mem-loads: false
      mem-stores: false
      minor-faults: true
      other_assists.any: false
      page-faults: false
      task-clock: false
  energy:
    <<: *base
    events:
      power/energy-cores/: false
      power/energy-pkg/: false
      power/energy-ram/: false
  fp:
    <<: *base
  derived:
    <<: *base
    metrics:
      ALGO-INT: true
      BR-MISS-RATIO: true
      CACHE-MISS-RATIO: true
      FP-SCALAR: true
      FP-VECTOR: true
      GFLOPS: true
      IPC: true
      MEM-BW: true
  uncore_imc:
    <<: *base
```

marconi_system.yaml

```
perf:
  enable: true
  generic:
    enable: true
  fp:
    enable: true
    fp_512d: true
    fp_512s: true
  derived:
    enable: true
```

Module responsible for querying processor-related counters using the perf tool.

hpcmd: memory configuration



default.yaml

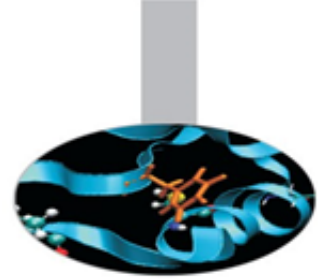
```
memory:
  <<: *base
  generic: # "numastat -m"
    <<: *base
    events:
      Active: false
      Active(anon): false
      Active(file): false
      AnonHugePages: false
      AnonPages: false
      Bounce: false
      Dirty: false
      FilePages: false
      HugePages_Free: false
      HugePages_Surp: false
      HugePages_Total: false
      Inactive: false
      Inactive(anon): false
      Inactive(file): false
      KernelStack: false
      Mapped: false
      MemFree: false
      MemTotal: true
      MemUsed: true
      Mlocked: false
      NFS_Unstable: false
      PageTables: false
      SReclaimable: false
      SUnreclaim: false
      Shmem: false
      Slab: false
      Unevictable: false
      Writeback: false
      WritebackTmp: false
  numa: # "numastat -s"
    <<: *base
    events:
      Interleave_Hit: false
      Local_Node: false
      Numa_Foreign: false
      Numa_Hit: true
      Numa_Miss: true
      Other_Node: false
  rss: # "ps -U user -u user -o rss"
    <<: *base
```

marconi_system.yaml

```
memory:
  enable: true
  generic:
    enable: true
  rss:
    enable: true
```

Module responsible for querying memory related counters.

hpcmd: network configuration



default.yaml

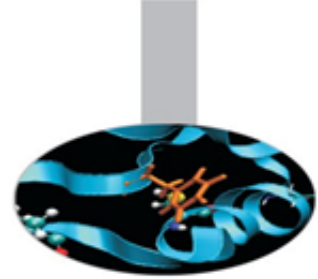
```
network:
  <<: *base
  ip: # "ip -s link show link_name"
    <<: *base
    link_name: ib0
    events:
      ib_faults: false
      rx_bytes: true
      rx_packets: true
      tx_bytes: true
      tx_packets: true
  ib: # read from port_path
    <<: *base
    port_path: /sys/class/infiniband/mlx5_0/ports/1/counters
    events:
      port_rcv_data: true
      port_rcv_packets: true
      port_xmit_data: true
      port_xmit_packets: true
  opa: # command = "/usr/sbin/opainfo"
    <<: *base
    events:
      rcv_data: true
      rcv_packets: true
      xmit_data: true
      xmit_packets: true
```

marconi_system.yaml

```
network:
  enable: true
  opa:
    enable: true
```

Module responsible for querying network related counters.

hpcmd: filesystem configuration



default.yaml

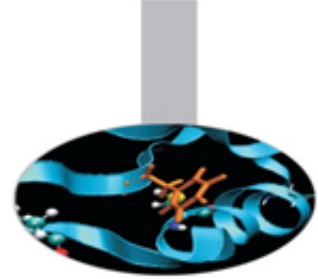
```
filesystem:
  <<: *base
  gpfs: # /usr/lpp/mmfs/bin/mmpmon -p -i cfg['filesystem']['gpfs']['cmd_path']
    <<: *base
    fs_names: all
    events:
      ip: false
      node: false
      rc: false
      timestamp: false
      tu: false
      cluster: false
      filesystem: false
      disks: false
      bytes_read: true
      bytes_written: true
      opens: true
      closes: true
      reads: true
      writes: true
      readdir: false
      inode_updates: true
  lustre:
```

marconi_system.yaml

```
filesystem:
  enable: true
  gpfs:
    fs_names: home,work,scratch
    enable: true
```

Module responsible for querying filesystem related counters

hpcmd: software configuration



default.yaml

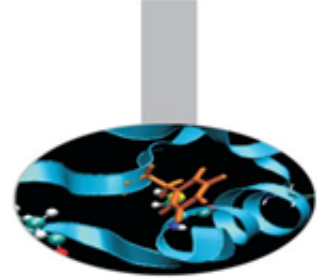
```
software:
  <<: *base
  executable:  # "ps -u {} --cols 256 -o cmd --sort -cputime".format(ps_exe_user)
    <<: *base
  libs:
    <<: *base
    default_paths:
      - /dev
      - /usr
      - /lib
      - /lib64
      - /run
      - /bin
      - /sbin
  threads: # "ps -U user -u user -L -opsr"
    <<: *base
    cores: 4
    sockets: 2
    combine_sockets: 1
  loadavg:  # "read /proc/loadavg
    <<: *base
    load_file:  /proc/loadavg
    events:
      load_1min: true
      load_5min: true
      load_15min:  true
      load_entities:  false
      load_PID:  false
```

marconi_system.yaml

```
software:
  enable:  true
  executable:
    enable: true
  libs:
    enable: true
  threads:
    enable: false
    cores: 48
  loadavg:
    enable: true
```

Module responsible for querying software information about the executable name and occupied/empty cores

hpcmd: system configuration



default.yaml

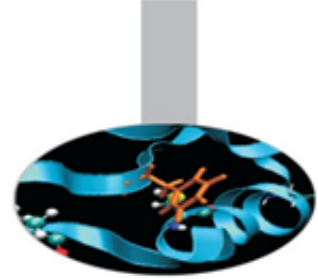
```
system:
  <<: *base
  services: # "ps -p {} -o cputime,rss".format(pid,)
    <<: *base
    events:
      slurmstepd: false
      systemd: false
```

marconi_system.yaml

```
system:
  enable: true
  services:
    enable: true
    events:
      slurmstepd: true
```

Module to trace the cpu time and memory usage of system processes.

hpcmd: physics configuration



default.yaml

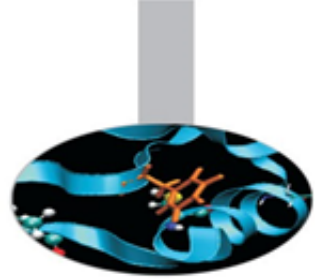
```
physics:
  <<: *base
  ipmi: # "ipmitool sdr type 'Current'"
    <<: *base
    lockfile: /tmp/load-sensor.pid
    events:
      current: false
```

marconi_system.yaml

```
physics:
  enable: false
  ipmi:
    enable: false
```

Module to trace the physical measurements from sensors of a node

III.1. hpcmd local metrics collection



Local metrics on
/var/log/hpcmd_stdout.log

SLURM batch job
[…]
#SBATCH –N 4
#SBATCH –p skl_sys_test
#SBATCH –qos=qos_test

SKL compute node
hpcmd daemon
/var/spool/slurmd
rsyslog.service

SKL compute node
hpcmd daemon
/var/spool/slurmd
rsyslog.service

SKL compute node
hpcmd daemon
/var/spool/slurmd
rsyslog.service

SKL compute node
hpcmd daemon
/var/spool/slurmd
rsyslog.service

HPCMD_CHECKPOINT="job_start" jobid="9169588" **nodeid="0"** userid="sbuenomi" opmode="systemd" epoch="60.0" awake="59.0" jobname="gmx-SKL-" jobstart="1616422533.33" nnodes="4" ntasks_per_node="24" ntasks="24" loadedmodules="profile/base:superc/2.0" cpus_per_task="2" threadspercore="None" realmemory="None" cores="48" sockets="2"

HPCMD_CHECKPOINT="job_start" jobid="9169588" **nodeid="1"** userid="sbuenomi" opmode="systemd" epoch="60.0" awake="59.0" jobname="gmx-SKL-" jobstart="1616422533.33" nnodes="4" ntasks_per_node="24" ntasks="24" loadedmodules="profile/base:superc/2.0" cpus_per_task="2" threadspercore="None" realmemory="None" cores="48" sockets="2"

HPCMD_CHECKPOINT="job_start" jobid="9169588" **nodeid="2"** userid="sbuenomi" opmode="systemd" epoch="60.0" awake="59.0" jobname="gmx-SKL-" jobstart="1616422533.33" nnodes="4" ntasks_per_node="24" ntasks="24" loadedmodules="profile/base:superc/2.0" cpus_per_task="2" threadspercore="None" realmemory="None" cores="48" sockets="2"

HPCMD_CHECKPOINT="job_start" jobid="9169588" **nodeid="3"** userid="sbuenomi" opmode="systemd" epoch="60.0" awake="59.0" jobname="gmx-SKL-" jobstart="1616422533.33" nnodes="4" ntasks_per_node="24" ntasks="24" loadedmodules="profile/base:superc/2.0" cpus_per_task="2" threadspercore="None" realmemory="None" cores="48" sockets="2"

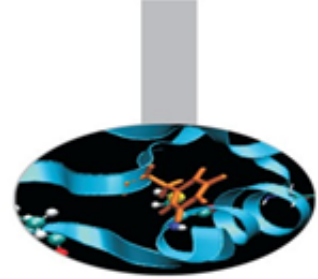
-- system wide prolog --
hpcmd_slurm_prolog.sh

```
HPCMD_TMP_DIR=/tmp/hpcmd
rm -f ${HPCMD_TMP_DIR}/suspend
rm -f ${HPCMD_TMP_DIR}/epilog
rm -fd ${HPCMD_TMP_DIR}
```

Local metrics on
/var/log/messages

Mar 22 15:16:00 r171c15s01 hpcmd.exe: HPCMD_CHECKPOINT="job_start" jobid="9169588" nodeid="0" userid="sbuenomi" opmode="systemd" epoch="60.0" awake="59.0" jobname="gmx-SKL-" jobstart="1616422533.33" nnodes="4" ntasks_per_node="24" ntasks="24" loadedmodules="profile/base:superc/2.0" cpus_per_task="2" threadspercore="None" realmemory="None" cores="48" sockets="2"

III.2. Data transport & collection configuration (I)



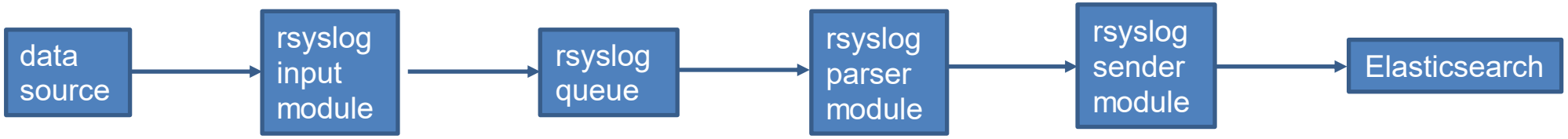
Local metrics:
/var/log/messages

rsyslog.con client configuration

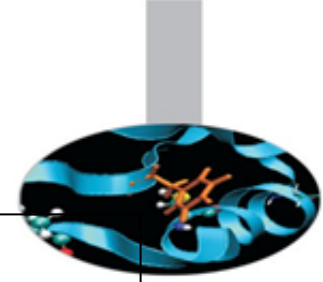
```
# send logs over TCP:
*.*  @@131.175.199.127:514
```

rsyslog.conf configuration

```
# modules for hpcmd:
module(load="imfile")
module(load="mmnormalize")
module(load="omelasticsearch")


if($msg contains 'HPCMD_CHECKPOINT')  then {

    if ($msg contains 'job_start') then {
        $template tpl-plain-syslog-hpcmd-js,  "%msg%\n"
        *.* /var/log/hpcmd_jobstart.log;tpl-plain-syslog-hpcmd-js
    }

    else if ($msg contains 'perf') then {
        $template tpl-plain-syslog-hpcmd-pf,  "%msg%\n"
        *.* /var/log/hpcmd_perf.log;tpl-plain-syslog-hpcmd-pf
    }

    else if …
}
```



III.2. Data transport & collection configuration (II)



hpcmd_jobstart.conf

```
input(type="imfile"
   File="/var/log/hpcmd_jobstart.log"
   ruleset="jobstart_output"
   tag="HPCMD_CHECKPOINT=\"job_start\""
)

# this is for Elasticsearch index names to be like "<index_name>-YYYY.MM.DD"
template(name="jobstart-index"
  type="list") {
   constant(value="jobstart-")
   property(name="timereported"  dateFormat="rfc3339"  position.from="1"
position.to="4")
   constant(value=".")
   property(name="timereported"  dateFormat="rfc3339"  position.from="6"
position.to="7")
   […]
}

# template to generate JSON documents for Elasticsearch:
template(name="all_json" type="list") {
   constant(value="{")
   constant(value="\"@timestamp\":\"")     property(name="timereported"
                                                    dateFormat="rfc3339")
   constant(value="\",\"host\":\"")        property(name="hostname")
   constant(value="\",\"nodeid\":\"")      property(name="$!nodeid"  format="json")
   constant(value="\",\"userid\":\"")      property(name="$!userid"  format="json")
   constant(value="\",\"opmode\":\"")      property(name="$!opmode"  format="json")
   constant(value="\",\"epoch\":\"")       property(name="$!epoch"  format="json")
   constant(value="\",\"awake\":\"")       property(name="$!awake"  format="json")
   constant(value="\",\"jobname\":\"")     property(name="$!jobname"  format="json")
   […]
   constant(value="\"}")
  }
```

```
ruleset(name="jobstart_output") {
   # parser:
   action(type="mmnormalize" rulebase="/opt/rsyslog/hpcmd_rule_job_start.rb")
   # sender:
   action(type="omelasticsearch"
      server="localhost"
      serverport="9200"
      template="all_json"
      searchIndex="jobstart-index"
      dynSearchIndex="on"
      searchType="events"
      bulkmode="on"
      queue.dequeuebatchsize="5000"
      queue.size="100000"
      queue.workerthreads="5"
     action.resumeretrycount="-1"
      errorfile="/var/log/omelasticsearch-jobstart.log"
   )
}
```

/opt/rsyslog/hpcmd_rule_job_start.rb

```
version=1
rule=:HPCMD_CHECKPOINT="job_start"  jobid="%jobid:number%"
nodeid="%nodeid:number%"  userid=%userid:quoted-string%
opmode=%opmode:quoted-string%  epoch="%epoch:float%"
awake="%awake:float%"  jobname=%jobname:quoted-string%
```