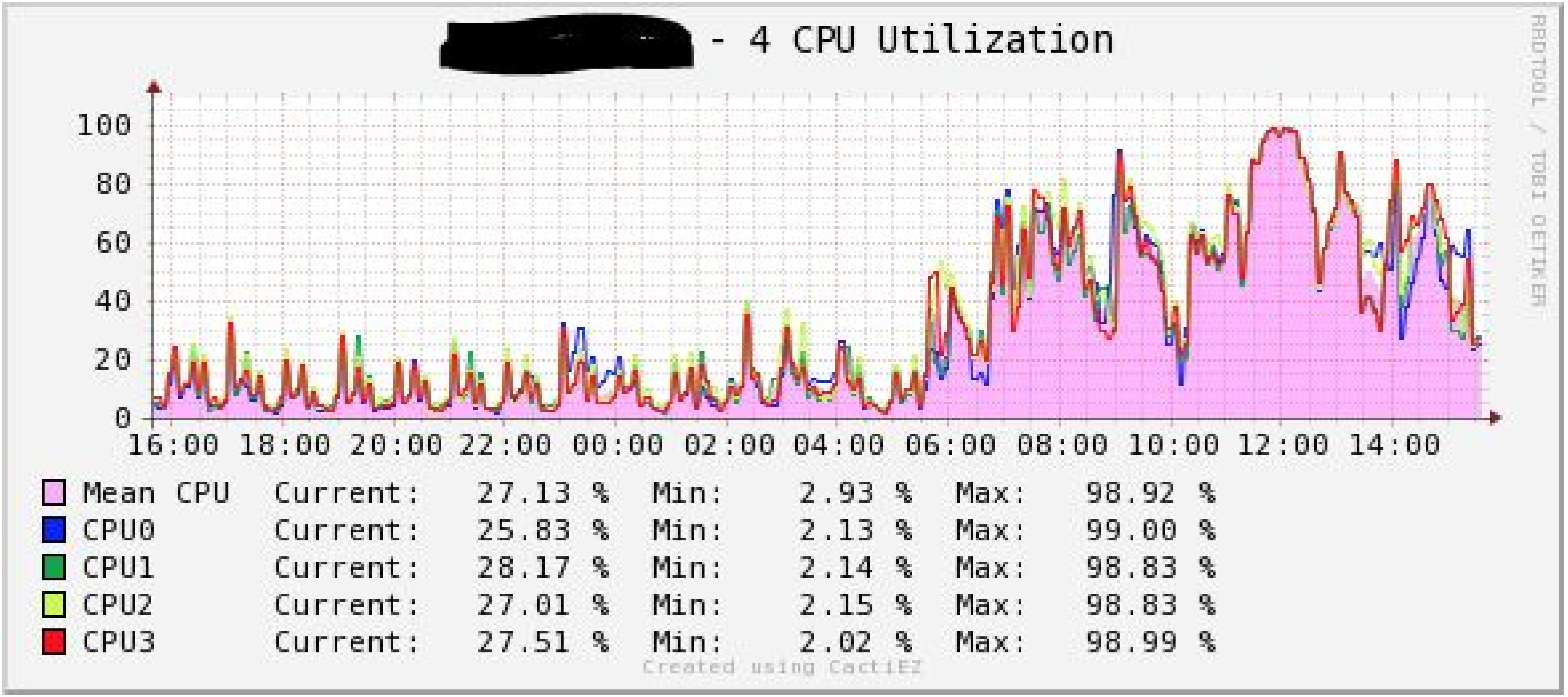
INFINITYWORKS

The Uninstrumentable; Getting Apache Spark and Prometheus to Play Nicely

DAN RATHBONE & JOE STRINGER

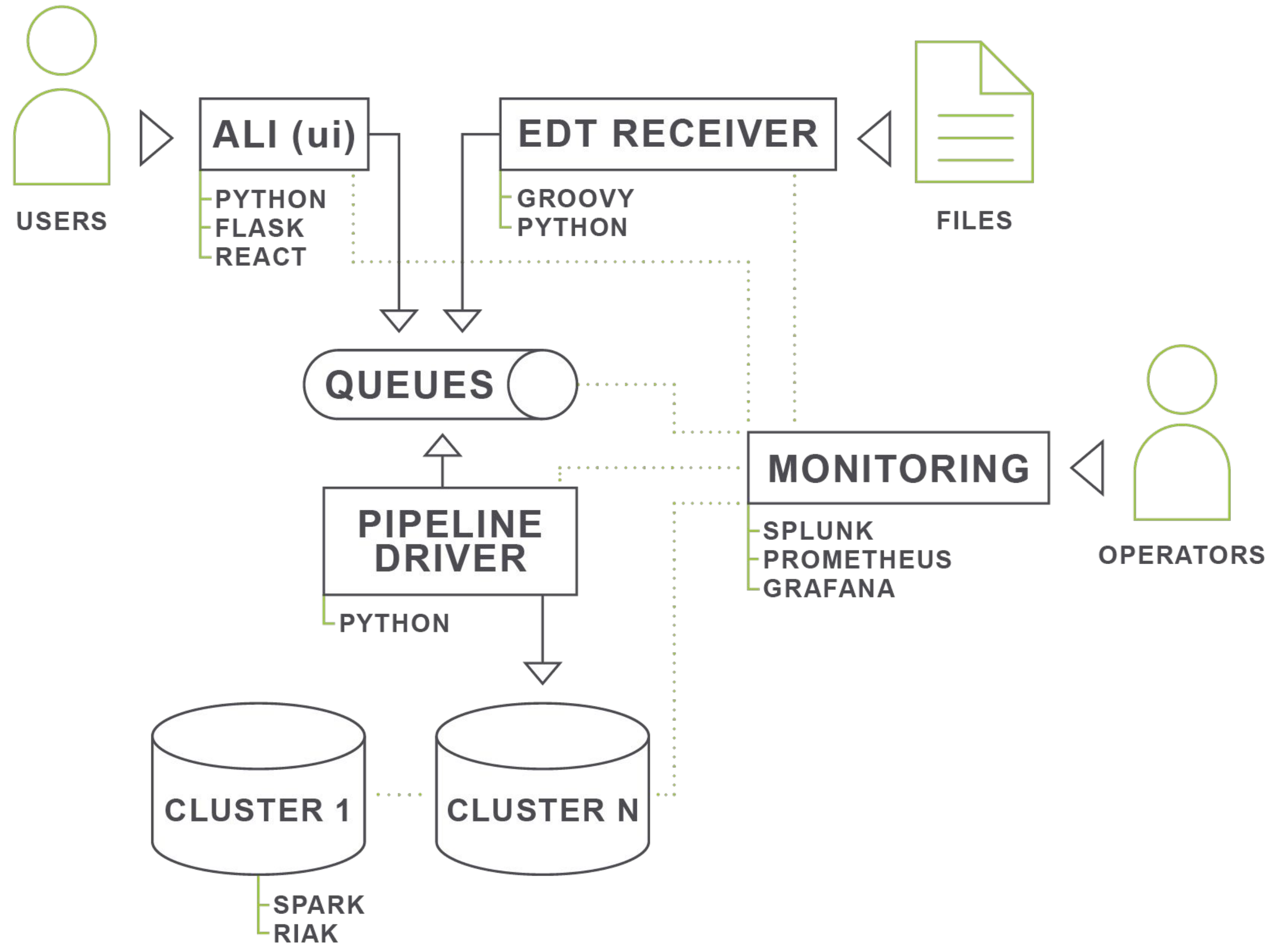
PROMCON 2017, AUGUST 2017

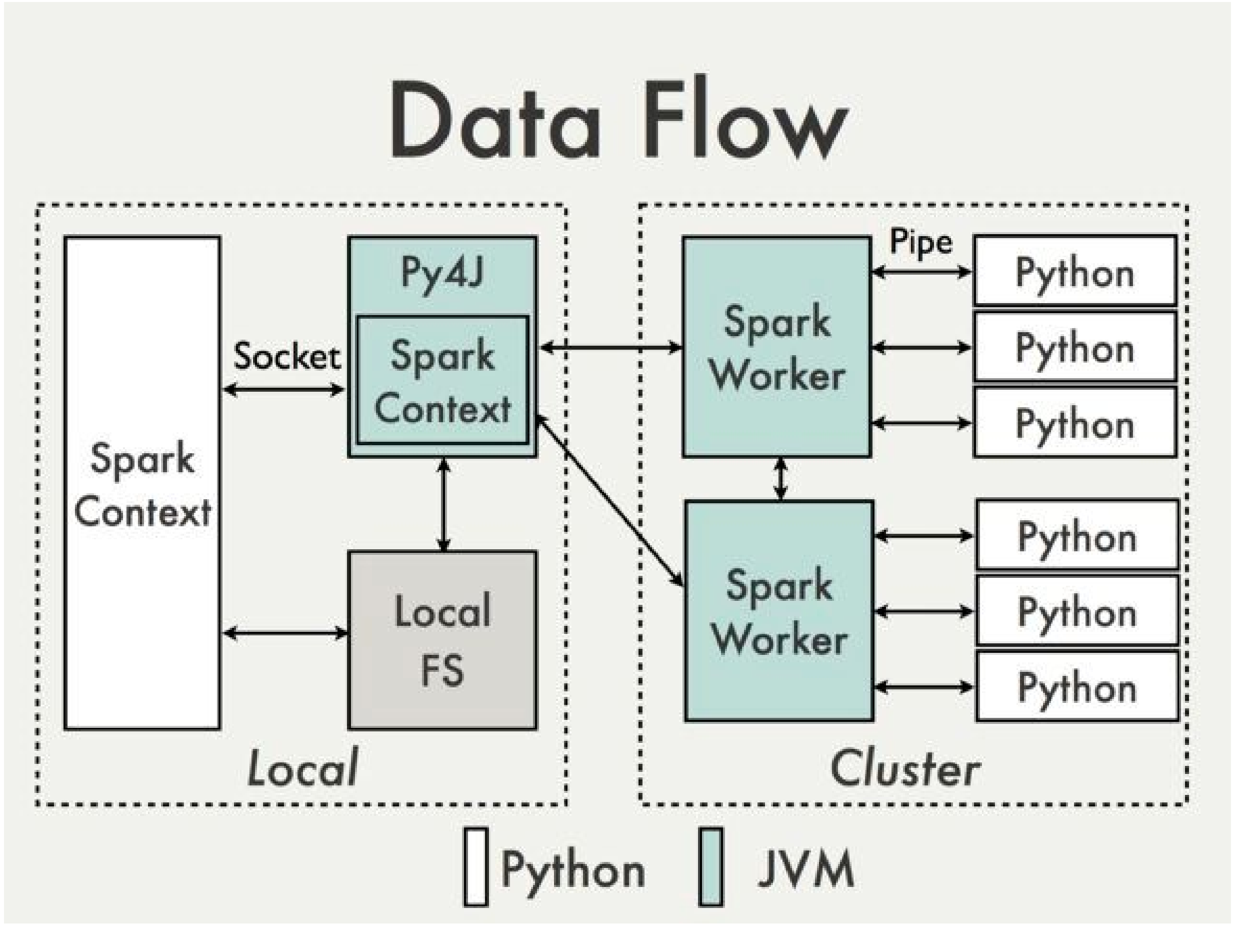
I'll just put this over here with the rest of the fire



```
🟪 Mean CPU   Current:   27.13 %   Min:    2.93 %   Max:   98.92 %
🟦 CPU0       Current:   25.83 %   Min:    2.13 %   Max:   99.00 %
🟩 CPU1       Current:   28.17 %   Min:    2.14 %   Max:   98.83 %
🟨 CPU2       Current:   27.01 %   Min:    2.15 %   Max:   98.83 %
🟥 CPU3       Current:   27.51 %   Min:    2.02 %   Max:   98.99 %
```

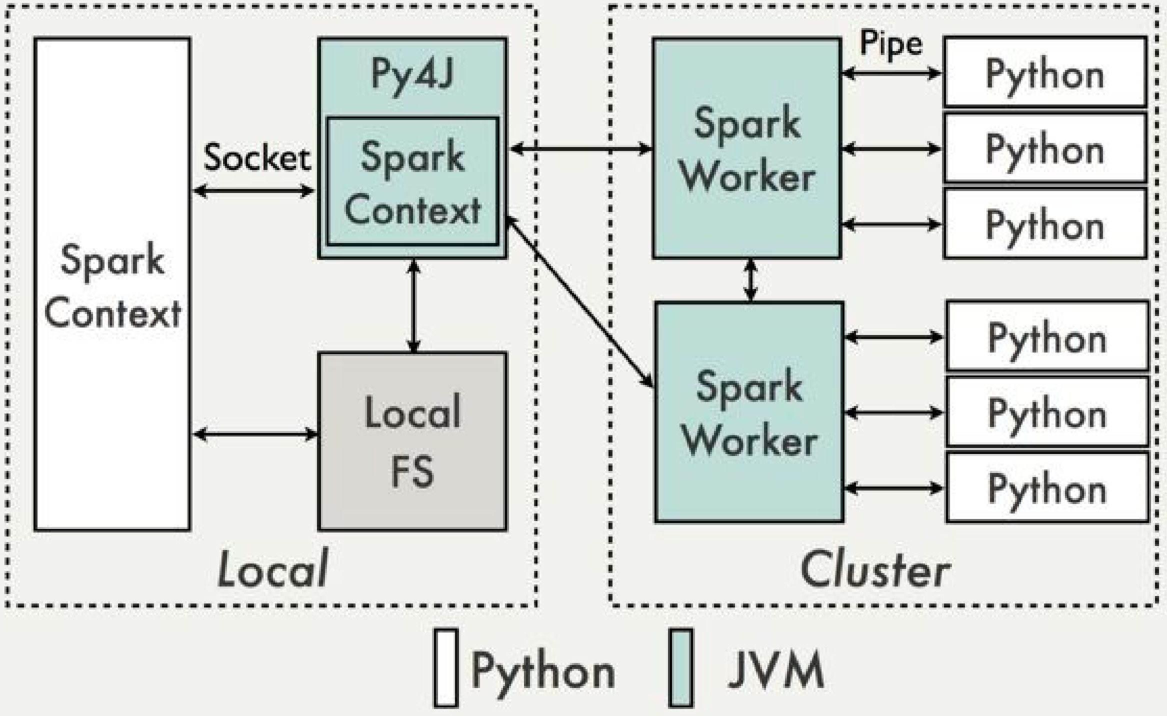
- Healthcare data processing system using Apache PySpark
- Failed attempts and the crazy ideas that followed
- Actually working with lots of pretty graphs


USERS
ALI (ui)
PYTHON
FLASK
REACT
EDT RECEIVER
GROOVY
PYTHON
FILES
QUEUES
PIPELINE DRIVER
PYTHON
MONITORING
SPLUNK
PROMETHEUS
GRAFANA
OPERATORS
CLUSTER 1
CLUSTER N
SPARK
RIAK



https://cwiki.apache.org/confluence/display/SPARK/PySpark+Internals

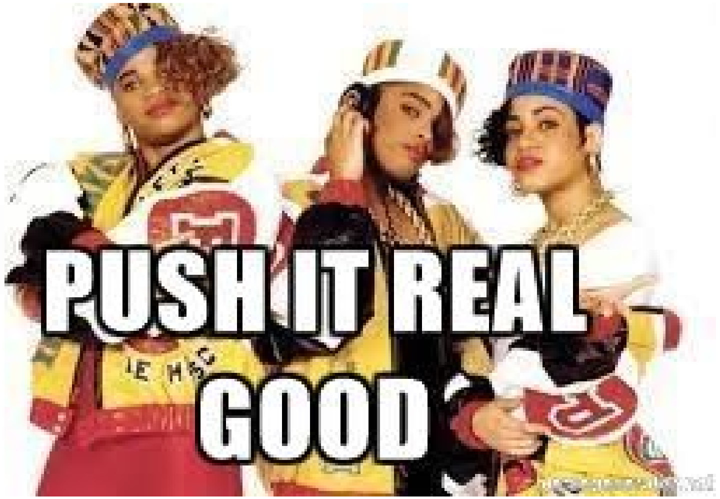
Data Flow

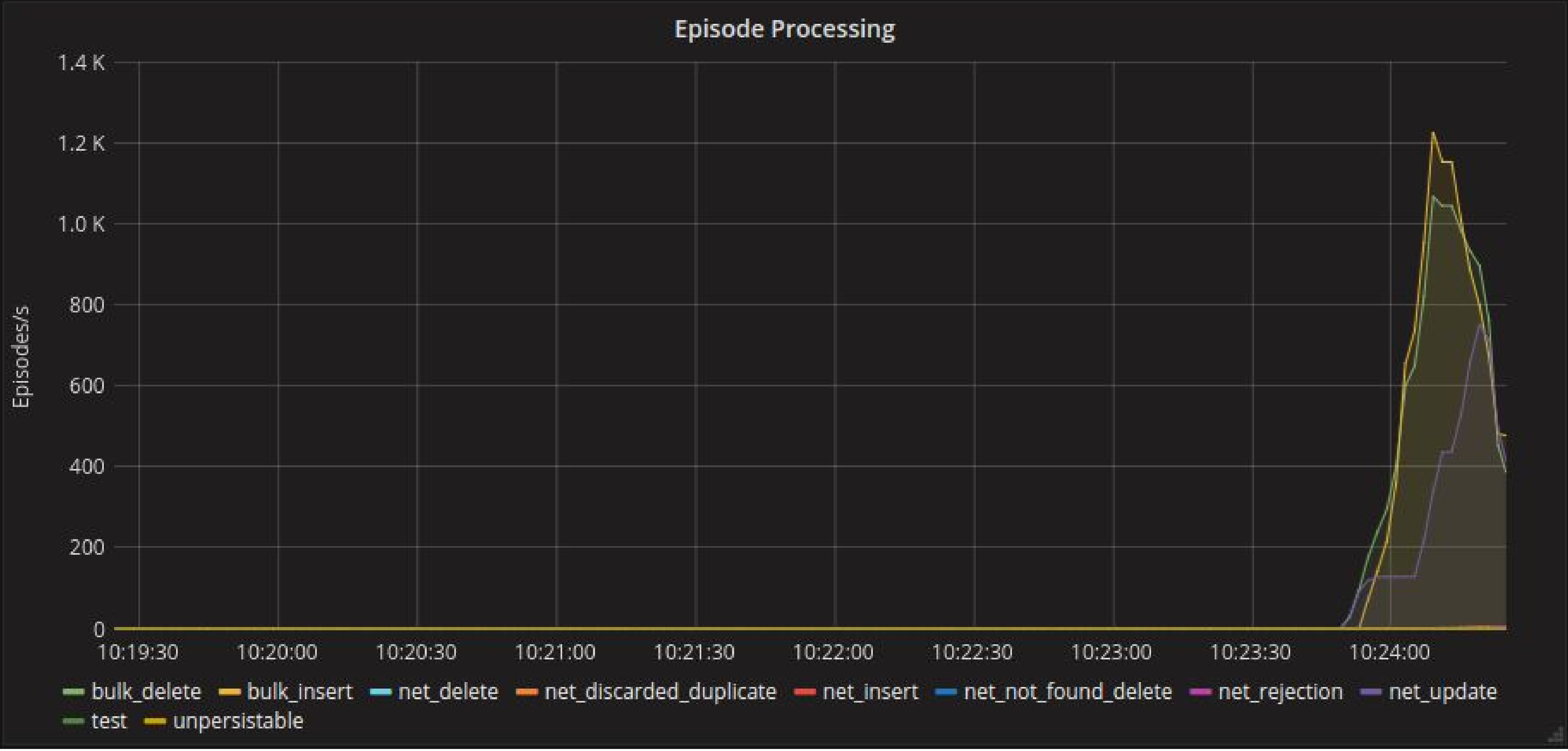


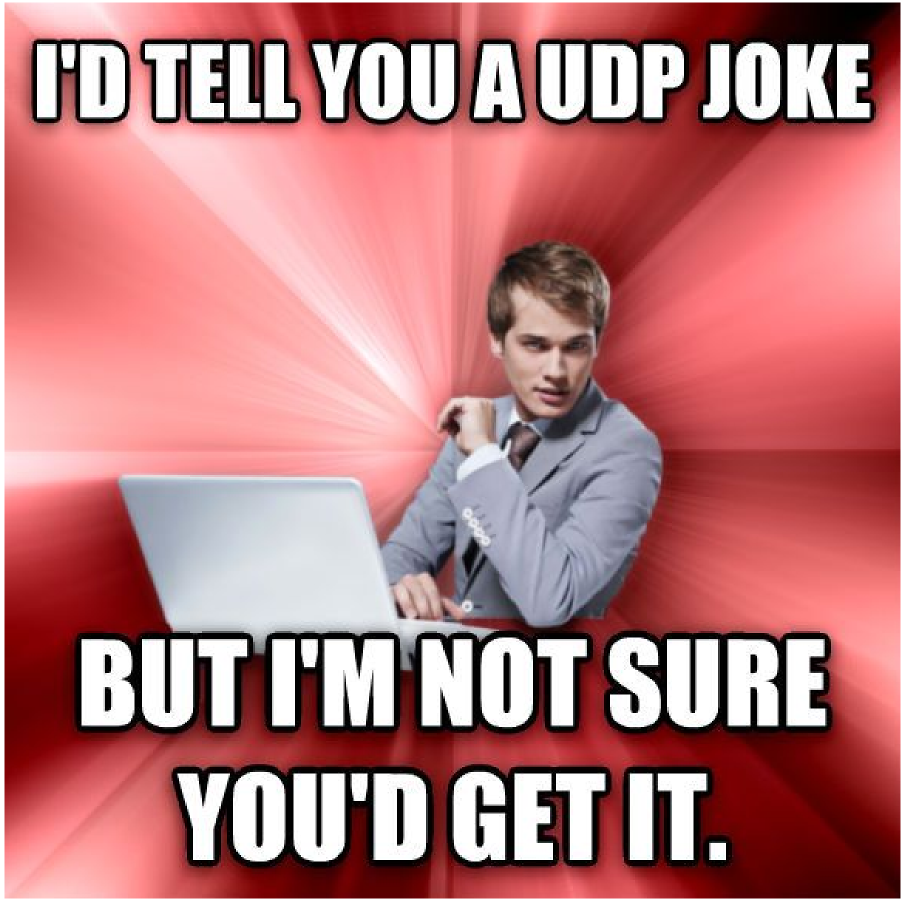
https://cwiki.apache.org/confluence/display/SPARK/PySpark+Internals

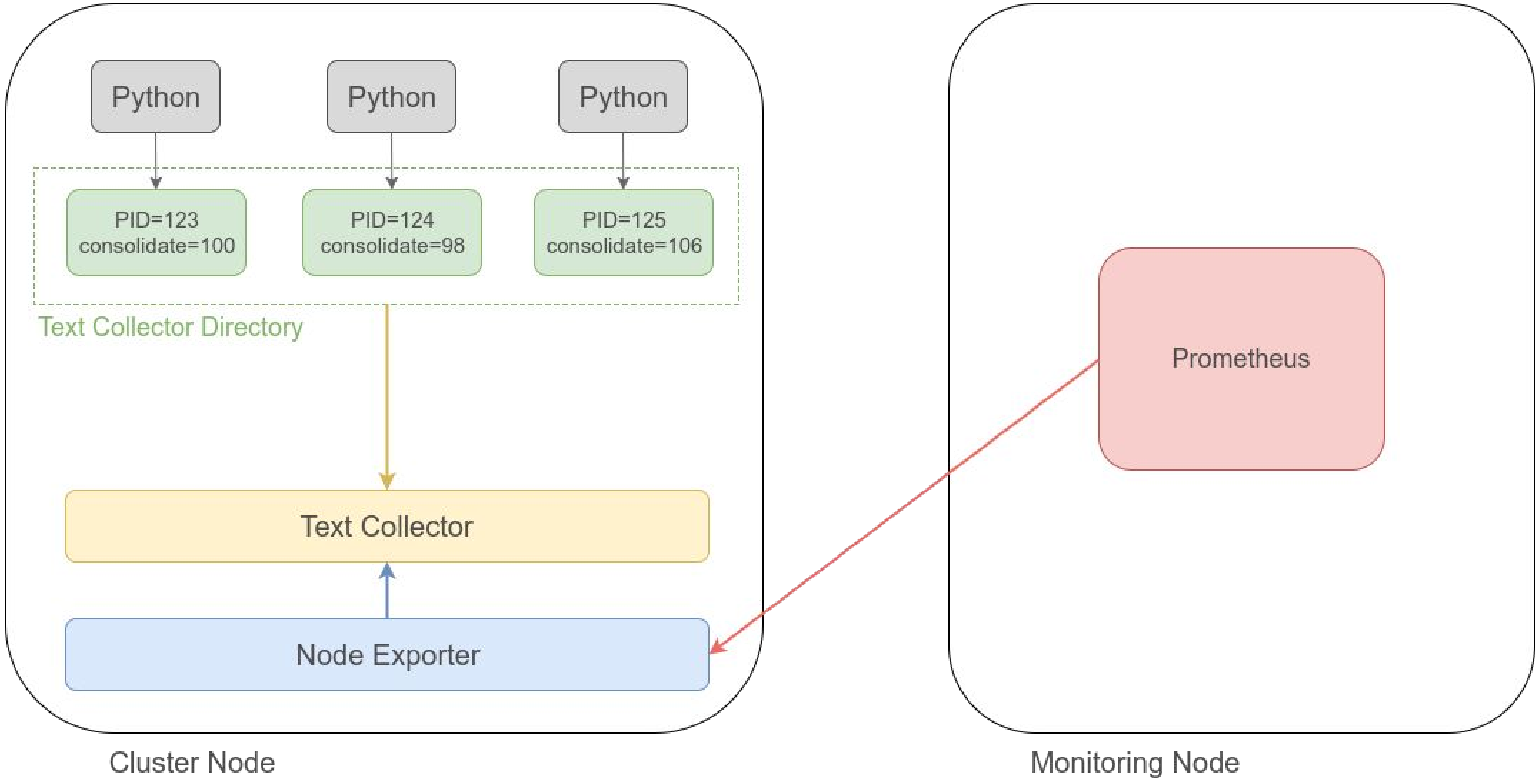
“Occasionally you will need to monitor components which cannot be scraped. They might live behind a firewall, or they might be too short-lived to expose data reliably via the pull model. The [Prometheus Pushgateway]() allows you to push time series from these components to an intermediary job which Prometheus can scrape.”

“The Pushgateway is explicitly not an aggregator or distributed counter but rather a metrics cache”



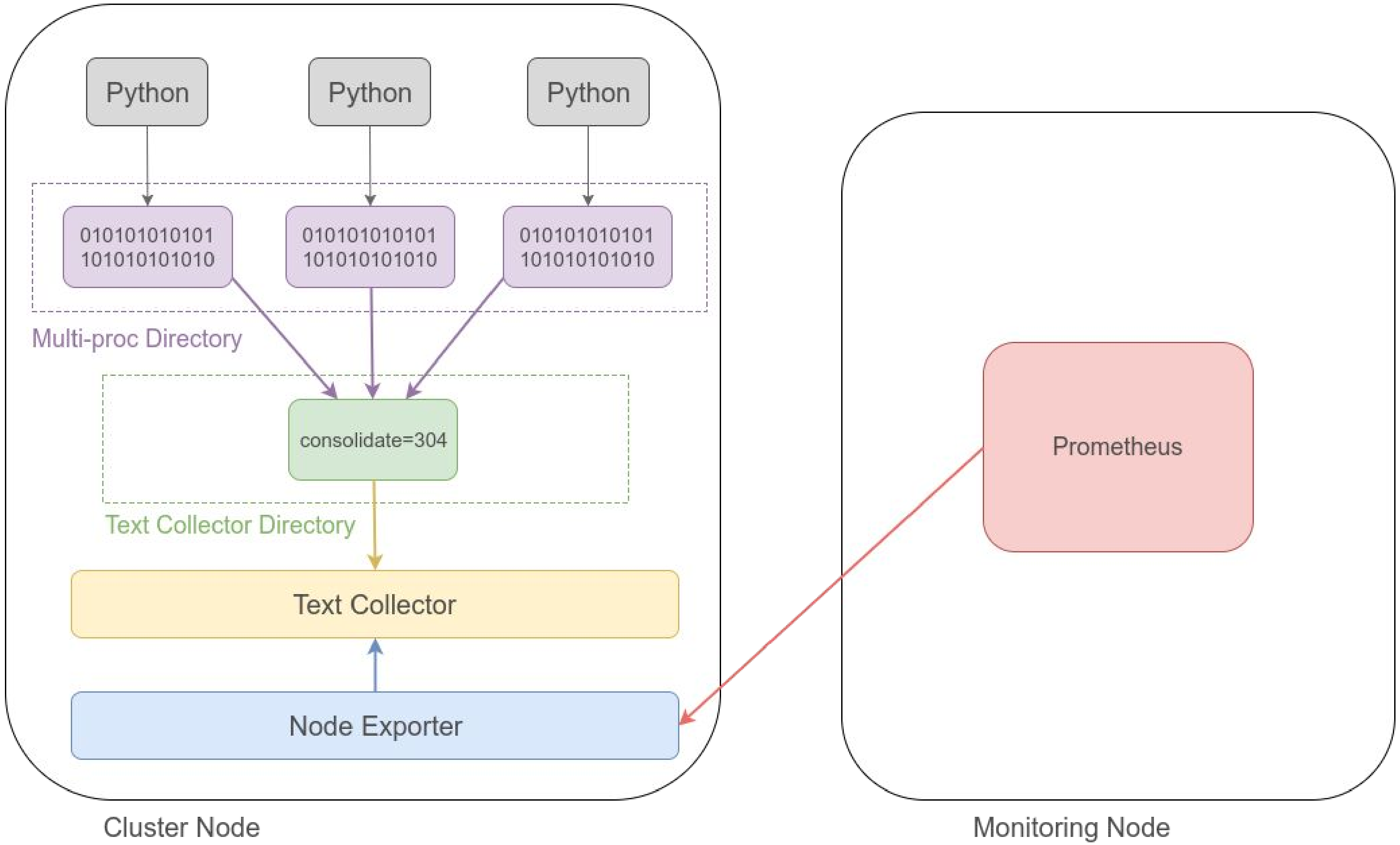

Episode Processing
Episodes/s
1.4 K
1.2 K
1.0 K
800
600
400
200
0
10:19:30
10:20:00
10:20:30
10:21:00
10:21:30
10:22:00
10:22:30
10:23:00
10:23:30
10:24:00
bulk_delete
bulk_insert
net_delete
net_discarded_duplicate
net_insert
net_not_found_delete
net_rejection
net_update
test
unpersistable


I'D TELL YOU A UDP JOKE
BUT I'M NOT SURE YOU'D GET IT.

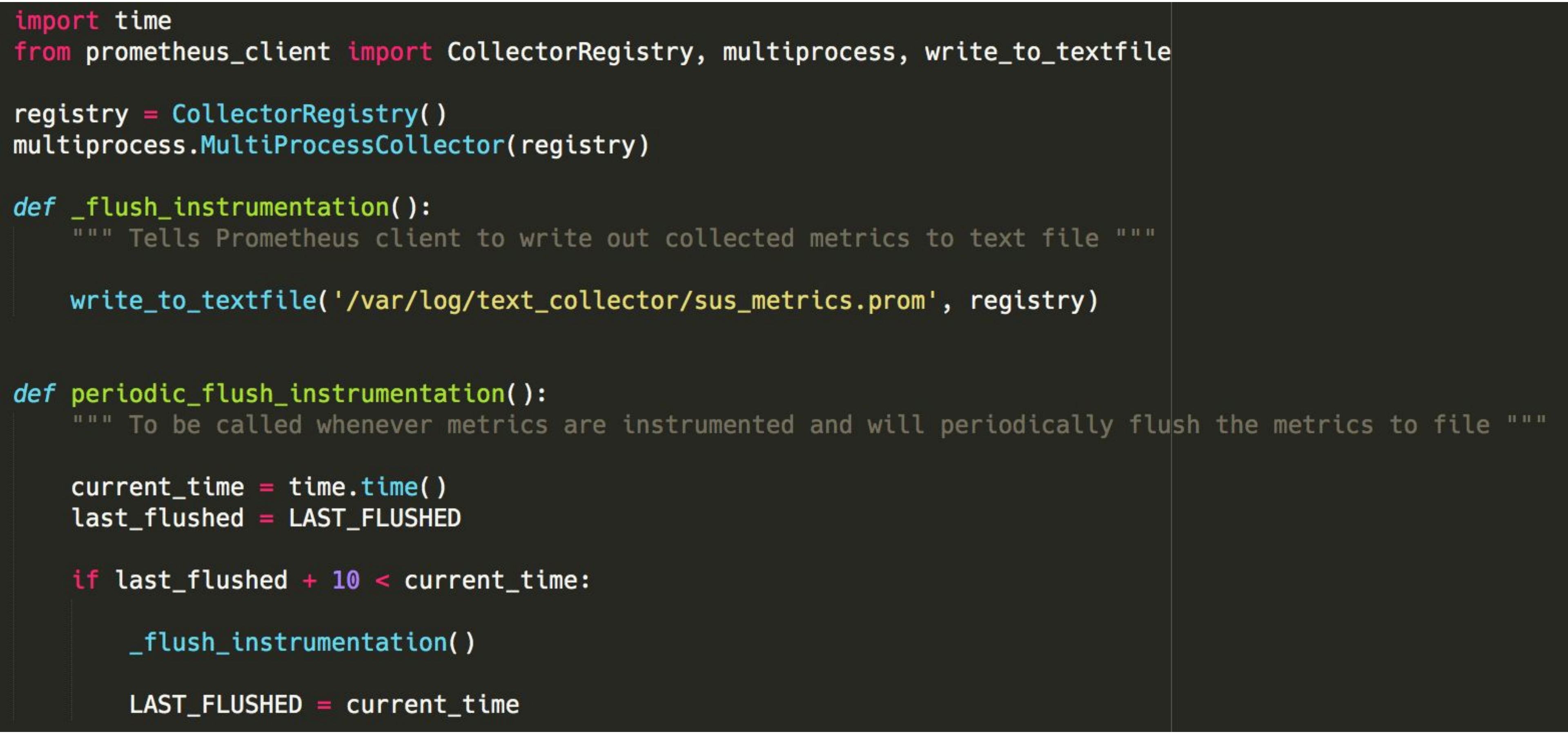

Python
Python
Python
PID=123 consolidate=100
PID=124 consolidate=98
PID=125 consolidate=106
Text Collector Directory
Text Collector
Node Exporter
Prometheus
Cluster Node
Monitoring Node

```scala
/**
 * Monitor all the idle workers, kill them after timeout.
 */
private class MonitorThread extends Thread(s"Idle Worker Monitor for $pythonExec") {

  setDaemon(true)

  override def run() {
    while (true) {
      synchronized {
        if (lastActivity + IDLE_WORKER_TIMEOUT_MS < System.currentTimeMillis()) {
          cleanupIdleWorkers()
          lastActivity = System.currentTimeMillis()
        }
      }
      Thread.sleep(10000)
    }
  }
}
```

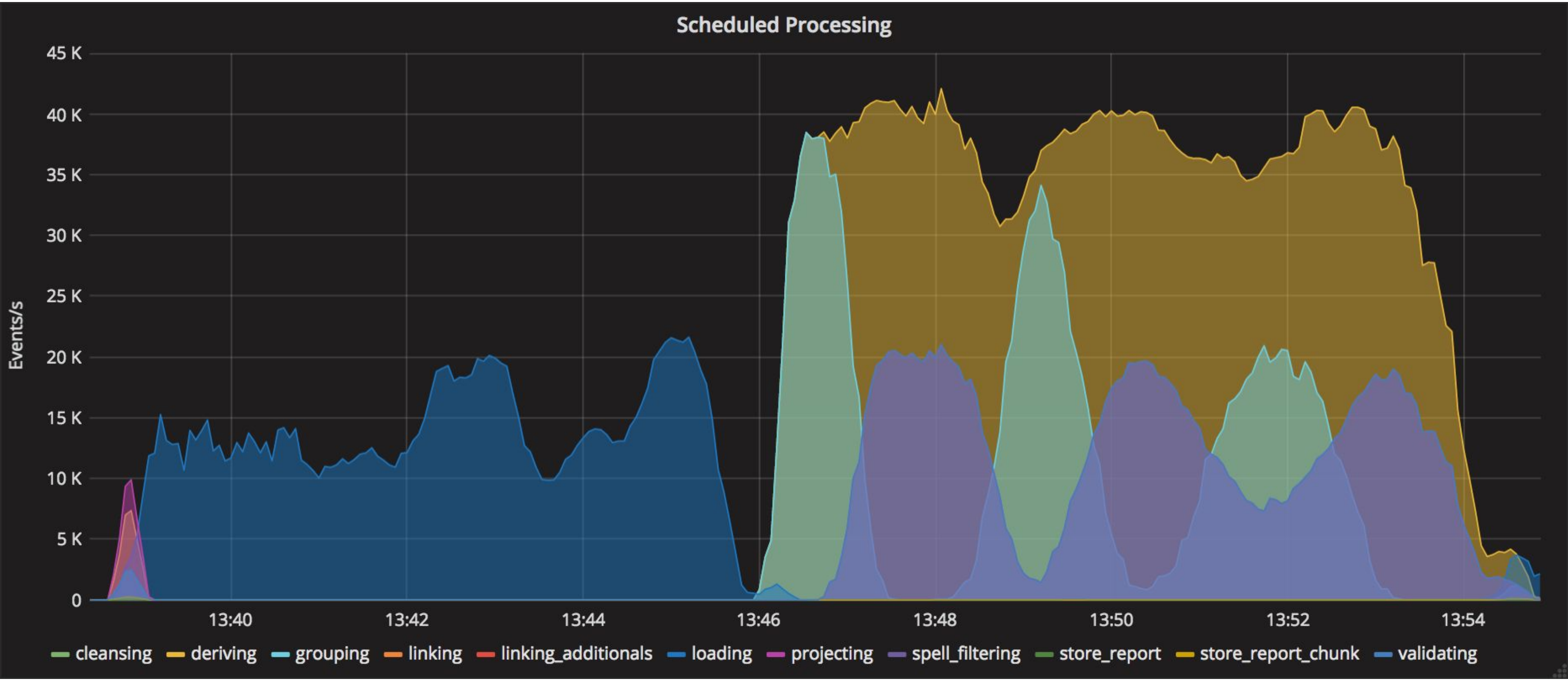
1 MILLION
TIMESERIES

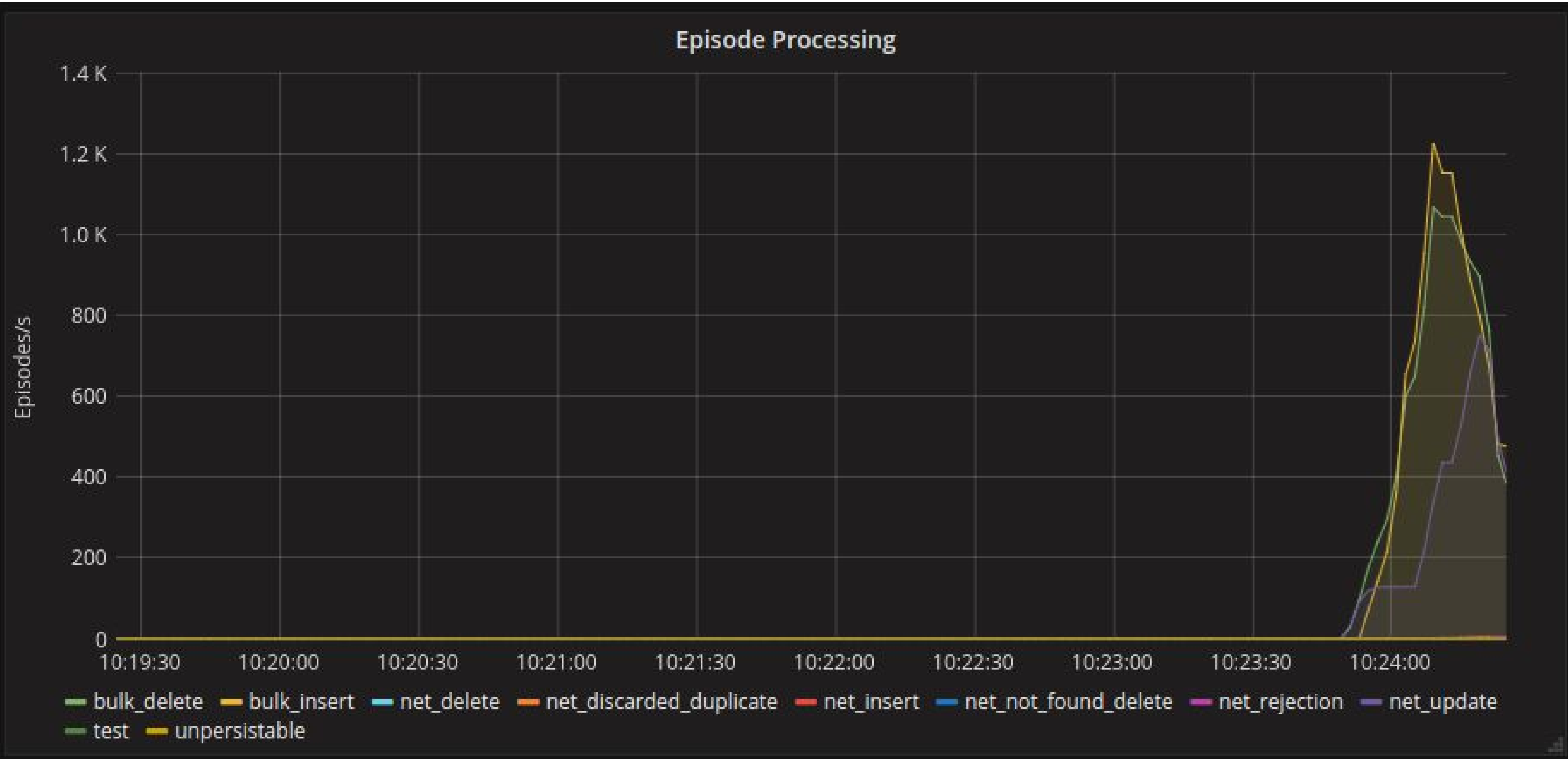

Python
Python
Python
010101010101
101010101010
010101010101
101010101010
010101010101
101010101010
Multi-proc Directory
consolidate=304
Text Collector Directory
Text Collector
Node Exporter
Cluster Node
Prometheus
Monitoring Node

```
PROM_CONSOLIDATION_COUNT.labels(type=type).inc(increment)

periodic_flush_instrumentation()
```

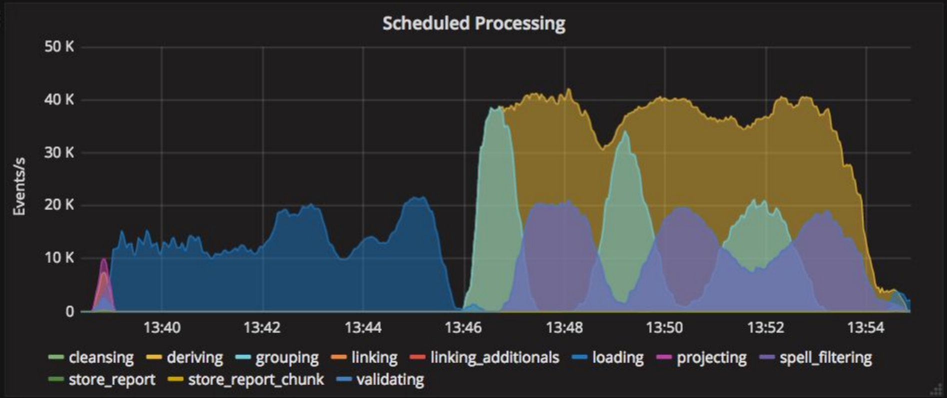


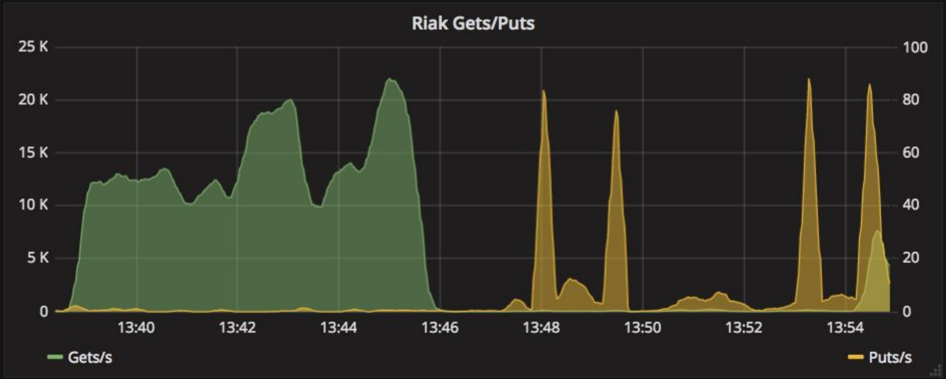
```python
import time
from prometheus_client import CollectorRegistry, multiprocess, write_to_textfile

registry = CollectorRegistry()
multiprocess.MultiProcessCollector(registry)

def _flush_instrumentation():
    """ Tells Prometheus client to write out collected metrics to text file """

    write_to_textfile('/var/log/text_collector/sus_metrics.prom', registry)


def periodic_flush_instrumentation():
    """ To be called whenever metrics are instrumented and will periodically flush the metrics to file """

    current_time = time.time()
    last_flushed = LAST_FLUSHED

    if last_flushed + 10 < current_time:

        _flush_instrumentation()

        LAST_FLUSHED = current_time
```

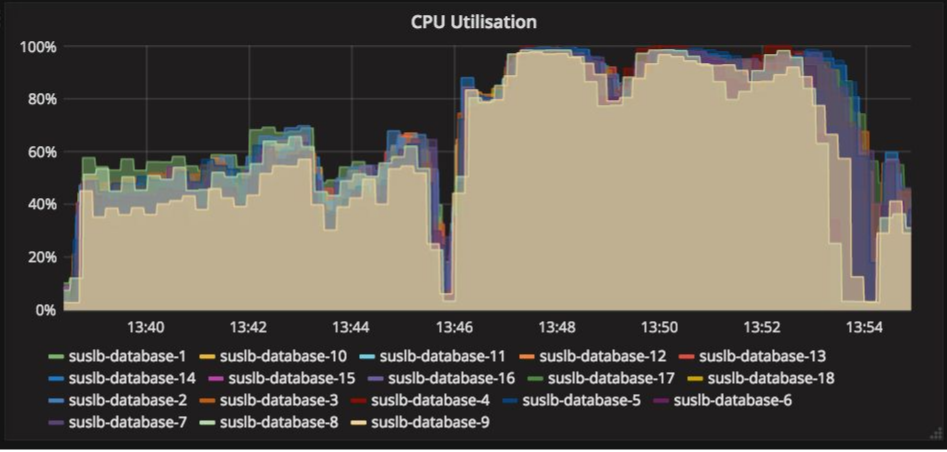

Scheduled Processing
Events/s
45 K
40 K
35 K
30 K
25 K
20 K
15 K
10 K
5 K
0
13:40
13:42
13:44
13:46
13:48
13:50
13:52
13:54
cleansing
deriving
grouping
linking
linking_additionals
loading
projecting
spell_filtering
store_report
store_report_chunk
validating

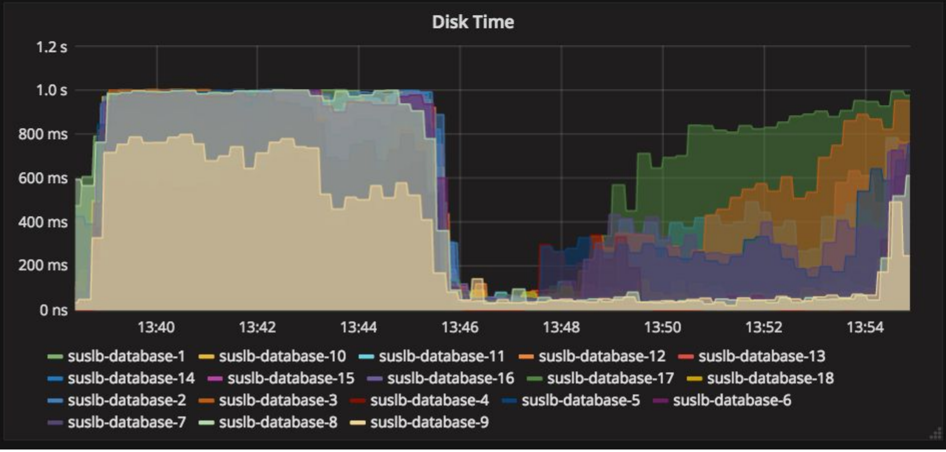


Episode Processing
1.4 K
1.2 K
1.0 K
800
600
400
200
0
Episodes/s
10:19:30
10:20:00
10:20:30
10:21:00
10:21:30
10:22:00
10:22:30
10:23:00
10:23:30
10:24:00
bulk_delete
bulk_insert
net_delete
net_discarded_duplicate
net_insert
net_not_found_delete
net_rejection
net_update
test
unpersistable


Scheduled Processing
Events/s
50 K
40 K
30 K
20 K
10 K
0
13:40
13:42
13:44
13:46
13:48
13:50
13:52
13:54
cleansing
deriving
grouping
linking
linking_additionals
loading
projecting
spell_filtering
store_report
store_report_chunk
validating


Riak Gets/Puts
25 K
20 K
15 K
10 K
5 K
0
100
80
60
40
20
0
13:40
13:42
13:44
13:46
13:48
13:50
13:52
13:54
Gets/s
Puts/s


CPU Utilisation
100%
80%
60%
40%
20%
0%
13:40
13:42
13:44
13:46
13:48
13:50
13:52
13:54
suslb-database-1
suslb-database-10
suslb-database-11
suslb-database-12
suslb-database-13
suslb-database-14
suslb-database-15
suslb-database-16
suslb-database-17
suslb-database-18
suslb-database-2
suslb-database-3
suslb-database-4
suslb-database-5
suslb-database-6
suslb-database-7
suslb-database-8
suslb-database-9


Disk Time
1.2 s
1.0 s
800 ms
600 ms
400 ms
200 ms
0 ns
13:40
13:42
13:44
13:46
13:48
13:50
13:52
13:54
suslb-database-1
suslb-database-10
suslb-database-11
suslb-database-12
suslb-database-13
suslb-database-14
suslb-database-15
suslb-database-16
suslb-database-17
suslb-database-18
suslb-database-2
suslb-database-3
suslb-database-4
suslb-database-5
suslb-database-6
suslb-database-7
suslb-database-8
suslb-database-9

- Realtime is worth the effort, visibility is key
- Nothing’s uninstrumentable
- The solution is often quite simple
- Prometheus is pretty flexible

DAN RATHBONE

@thetrilemma

JOE STRINGER

@joeds13

INFINITYWORKS