Monitoring HTCondor: A common one-stop solution?

Iain Steers - CERN IT
Introduction

This presentation is going to cover the various different ways of monitoring a HTCondor pool.

In an effort to solidify the best practices and build up a community around a common monitoring solution.
A Brief History of Monitoring HTCondor

We’ve all sat down and thought - ”What about the clis and unix cron?”.

Figure: One Hour Later: Why is RecentDaemonDutyCycle so high?
A Slight Improvement

Autoformat to the rescue!

Figure: A bit better, but only something you want a human to use.
Requests from Management

Figure: SCHEDD_COLLECT_STATS_BY_{BY,FOR} aggregate attributes from job matching expressions.
The Ganglia Integration

Push standard and custom metrics into a ganglia instance using the daemon GANGLIAD.

Figure: Ganglia Web Front-end
Ganglia Competition?

GangliaD is great!
But what about large sites which already have centralized monitoring solutions?

• ELK Stack
• Influxdb and Grafana
• sysdig
MetricsD

*Todd’s Talk of Lies* from Barcelona mentioned an incoming MetricsD.

Same metric configuration language but instead of just sending to ganglia, publishes json blobs with the monitoring samples.
MetricsD Example

```
[  
  Name    = "Availability";
  Value   = int(ifThenElse(IsCritical is undefined,
                           (RecentDaemonCoreDutyCycle < .95) ||
                           (FileTransferFileReadLoad_5m < 2.0) ||
                           (FileTransferFileWriteLoad_1m < 2.0),
                           !IsCritical));
  Desc    = "Average availability of CE";
  Scale   = 100;
  Units   = ";
  TargetType = "Scheduler";
]
```
As a site, CERN already has a centralized monitoring set-up, based on ElasticSearch/Kibana. Leaving the old gangliad a bit redundant.

Pull out interesting metrics from classads/jobs using python-bindings.

So we’ve found ourselves using a mix of existing solutions.
Our Main Dashboard

Figure: The main pool health dashboard
Our Draining Dashboard

Figure: Track the multi-core draining of the pool and wasted cpu over time.
Our Cgroups Monitoring

Figure: Cgroups Monitoring of Jobs
Too Many Dashboards

Too Many Dashboards and different Systems. (ELK/Influx/Grafana/Spark/Jupyter)

#MonitoringSucks
The Problem

This doesn’t seem to be a problem just at CERN.

Looking at the HTCondor community it seems everyone has done their own thing for monitoring.

Siloing knowledge, implementations and best practices.
Conclusion

Could we do better?

Could we bring together a community around monitoring HTCondor?

Get the Python, HTCondor experts and data junkies in a (virtual-)room together and come up with a common platform, to really reveal the health of your pool?
Questions?

Any Questions?