FreeSWITCH Monitoring

ClueCon, August 2016

Moisés Silva <moy@sangoma.com>
Manager, Software Engineering

We're Hiring

- Linux developers C/C++ or Python
- Anywhere in the world, paid relocation to Toronto or full time remote opportunities
- Fun and relaxed work environment

Agenda

- Monitoring Basics
- Metrics and Logs
- Alerting
- Traditional Solutions
- Emerging Solutions
- Afterword (FreeSWITCH Oxidized)

Monitoring Basics

- Monitoring is about knowing what is happening on your systems. The good and the bad
- Monitoring helps you with prediction/ forecasting. Plan your future growth and anticipate problems
- Monitoring provides you with the data you need to troubleshoot problems faster

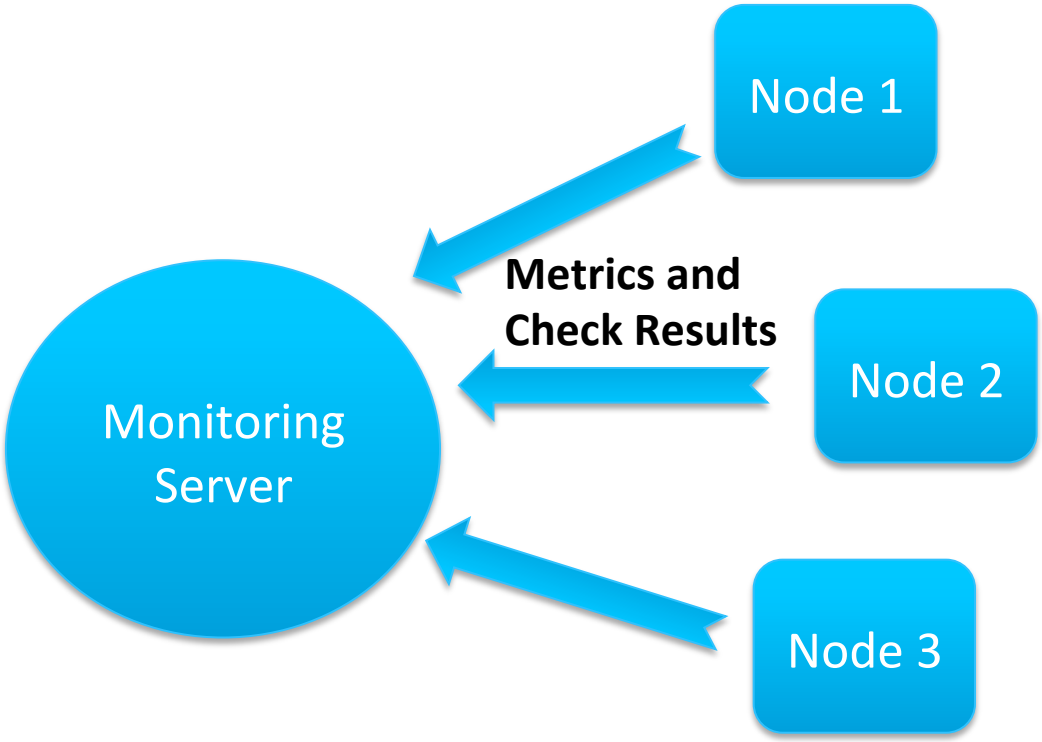
Types of Monitoring

- Application Metric Monitoring
  - Numbers about application usage & performance
  - You can then use those numbers to define alerts
- Service Status Monitoring
  - Simple binary checks. It's the service up or not

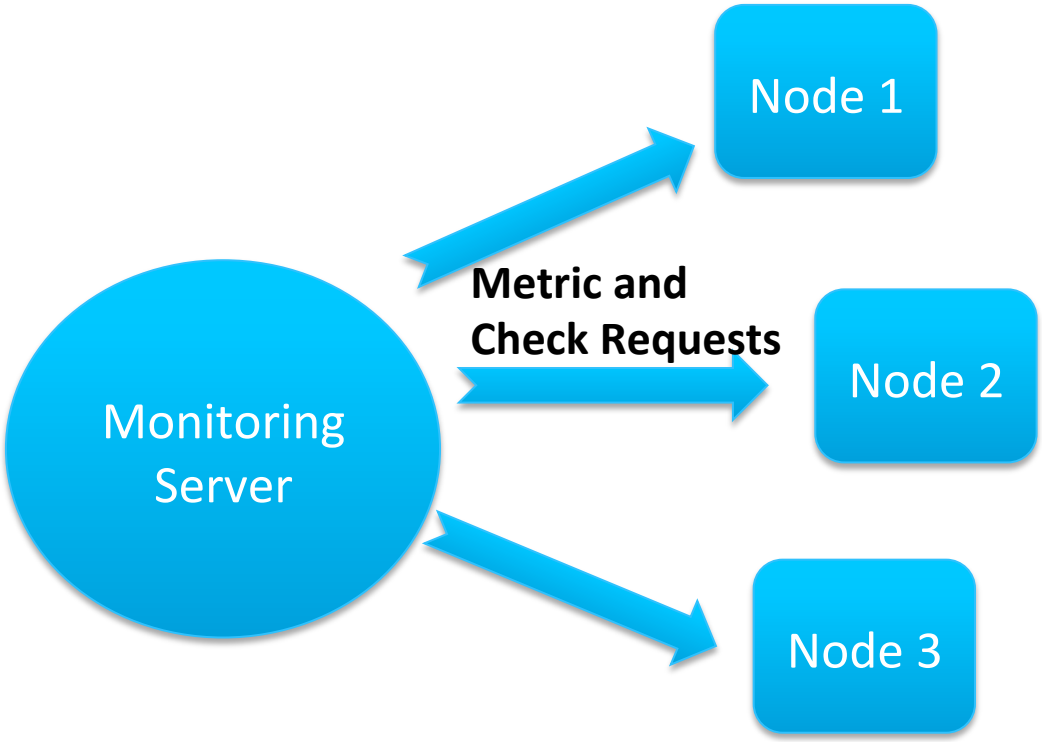
Monitoring Models

- Push vs Pull systems
  - Push: Graphite, StatsD, CollectD, SNMP
  - Pull: Prometheus, SNMP
- Determines who initiates the metric transfer
- Push systems are more dynamic and tend to require less maintenance
- Pull systems require a source of truth and node discovery
- Pulling however allows you to more effectively detect if a node or application is down

Push Monitoring



Pull Monitoring



Metrics

- Time series data. Sequence of numerical data points listed in time order, usually sampled at regular intervals.
- Different granularities of data stored and aggregated over time on fixed-size storage
- Time series analysis over the data results in fancy graphs
- Example databases: RRDTool, Graphite (Whisper), InfluxDB, Prometheus

FreeSWITCH Metrics

- Active Calls
- Total Calls
- Failed Calls
- Registrations
- Failed Registrations

FreeSWITCH Metrics

- ASR (Answer / Seizure Ratio)
- NER (Network Effectiveness Ratio)
- ACD (Average Call Duration)
- PDD (Post Dial Delay)

Logs

- Logs can be a source of metrics when metrics and service checks are not enough
- You may want to send alerts based on log patterns and repeated log errors
- ELK Stack
  - Elastic Search
  - Logstash
  - Kabana

Alerting

- Email, Chat (Slack/HipChat), SMS, etc
- Alert based on expected metrics and check failures
- Procure to use percentiles instead of averages
- Several alternatives: Prometheus, Alerta, Flapjack

Alert Fatigue

- This happens when alerts are triggered too often and causes the receiver to be desensitized

Alert Fatigue

- You know you have it when you start ignoring alerts
- Ignoring alerts leads to missing real problems or taking too long to respond to them

Alert Fatigue

- Warnings might be well intended and in isolation work well, however, when they add up, they may cause more harm than good

- Adjust your thresholds. Be cautious with warning vs critical severity and adjust your notifications accordingly

- Kill or fix alerts that are firing too often. You need a way to easily mute them in the meantime

Traditional Solutions

- They're all about getting the job done. And they do:

- But at some point you gotta ask yourself if there's a better way ...

SNMP

SNMP

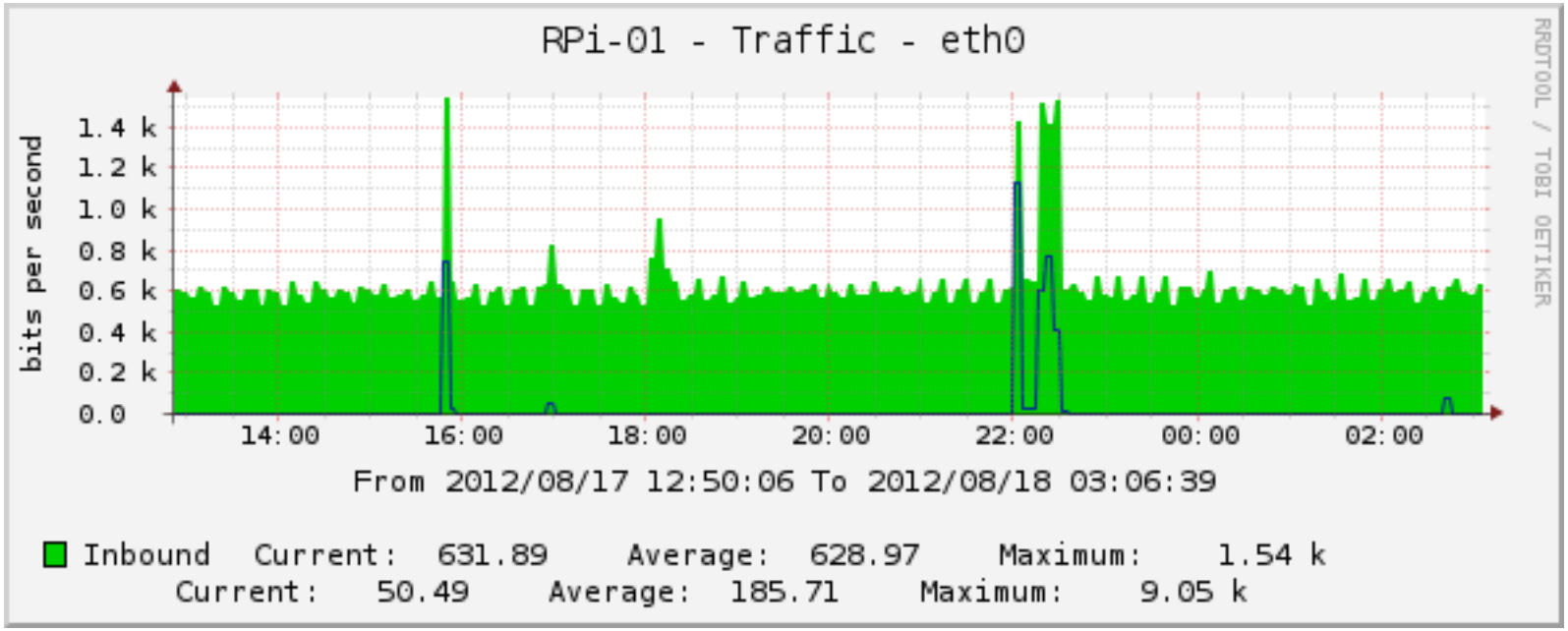
- MIBs/OIDs are cumbersome to use (to say the least)
  - OIDs available in FreeSWITCH give you number of sessions, active calls, sessions per second etc.
- Not easy to collect custom metrics
- Very limited information exposed by mod_snmp and overall seems kind of abandoned
- The monitoring/management interface in FreeSWITCH needs some work

CACTI & RRDtool

- Web front end for the RRDtool time-series database
- Collect metrics from multiple sources (e.g snmp)
- No one can ever remember how to add a server (convoluted process)

CACTI & RRDtool

- You end up with ugly graphs like this:



It’s better than nothing though ….

Monitoring Sucks

- So much that around 2011 a “Monitoring Sucks” community was set to fix it:
  - https://github.com/monitoringsucks
  - https://github.com/monitoringsucks/blog-posts
  - https://vimeo.com/monitorama

New Solutions

- Because new is always better …

Sensu

- Monitoring that doesn't suck (their motto)
- You can reuse nagios/zabbix checks. Sensu was designed as a replacement for aging Nagios installations
- No time-series database included, you have several options available (e.g Graphite, InfluxDB)

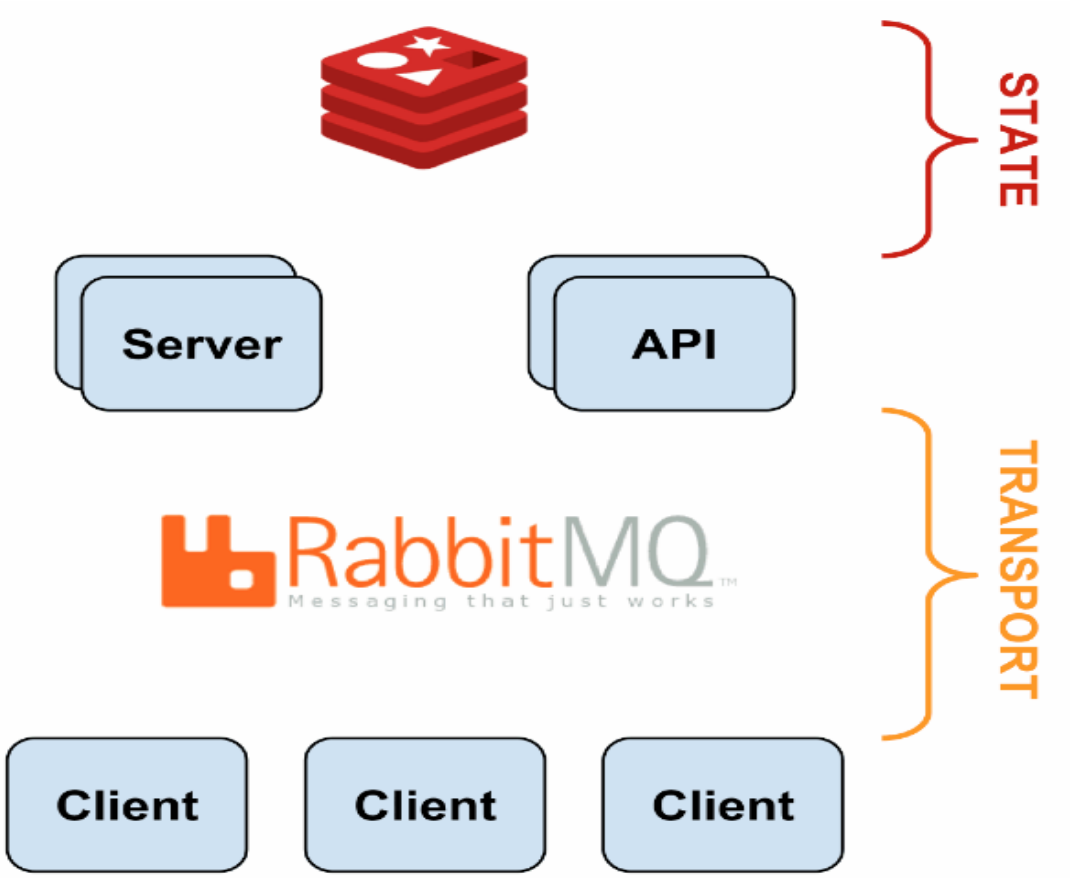
Sensu

- Pull/Push system with an agent
- Pub/Sub model with pluggable transports (RabbitMQ, Redis, Amazon SQS)
- Nodes subscribe to groups of checks. The checks are scheduled by the server or the node.
- Both service and metric checks supported
- Check results can be sent asynchronously by external jobs via raw TCP + JSON (support in a github PR for HTTP)

Sensu

- HTTP API to read access nodes, checks, silence alarms, etc
- Nice and simple dashboard UI
- Dynamic infrastructure. Servers add themselves to the monitoring system.
- Composable json configuration for easy automation (e.g Ansible, SaltStack, Chef etc)

Sensu



Source: sensuapp.org

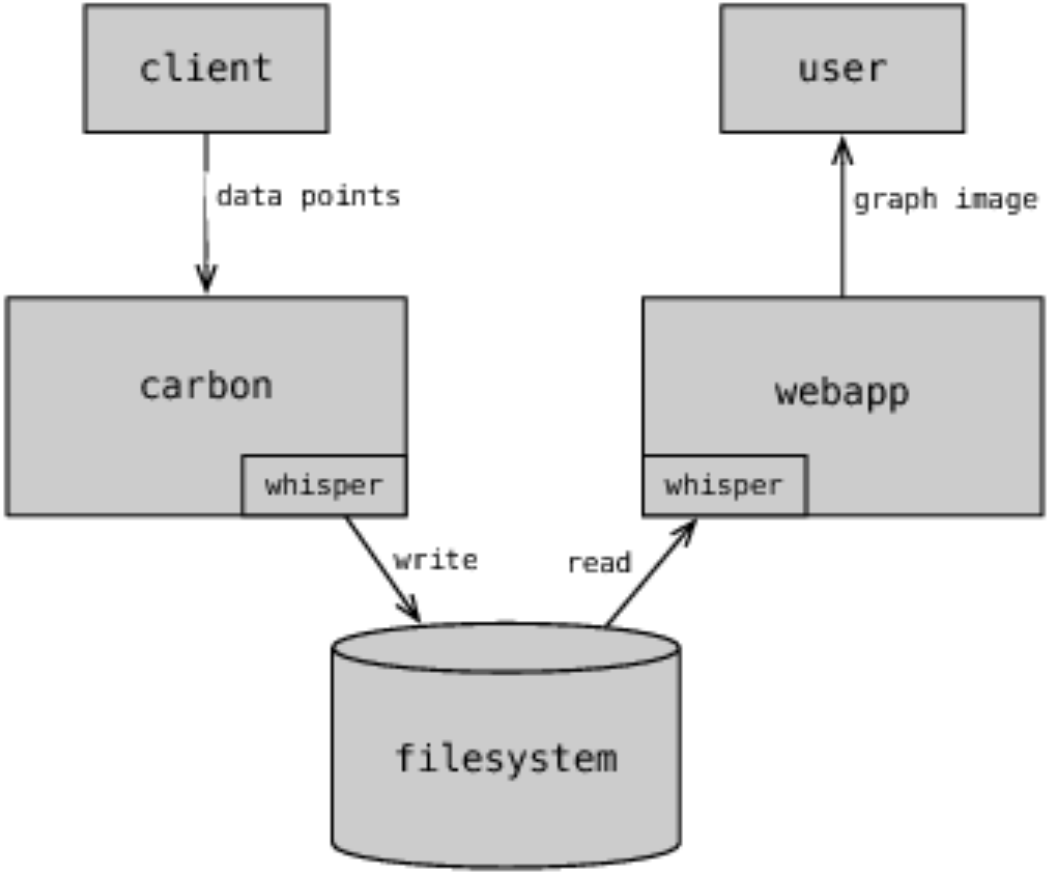
Graphite

- Time-series storage system
- Whisper database format designed as an improvement over RRDtool to manage out of time data points
- Carbon-cache is a daemon listening for metrics on a TCP socket (typically used in tandem with StatsD)
- Metrics are stored in fixed-size files
- Comes with a (quite outdated) UI for metric graphs. This is most often used as an http endpoint by other graphing tools like Grafana

Graphite

- Every time series is identified by a hierarchy of dot-separated identifiers (e.g stats.freeswitch.total_calls)
- The text format to send stats to graphite is dead simple: <metric name> <value> <timestamp>
  - stats.freeswitch.total_calls 50 1470843323
- No further dimensions can be encoded. The typical work-around is to encode them in the metric name:
  - stats.freeswitch.profile.internal.total_calls 20 1470843323

Graphite



Source: http://www.aosabook.org/en/graphite.html

Prometheus

- New time-series monitoring system with built-in alerting
- Includes its own time-series database format
- Multi-dimensional data model (fancy term for key/value pairs labels attached to the time-series metrics)
- HTTP-based pull model (Prometheus scraps nodes for metrics via HTTP)
- Push gateways supported for applications without a native HTTP endpoint

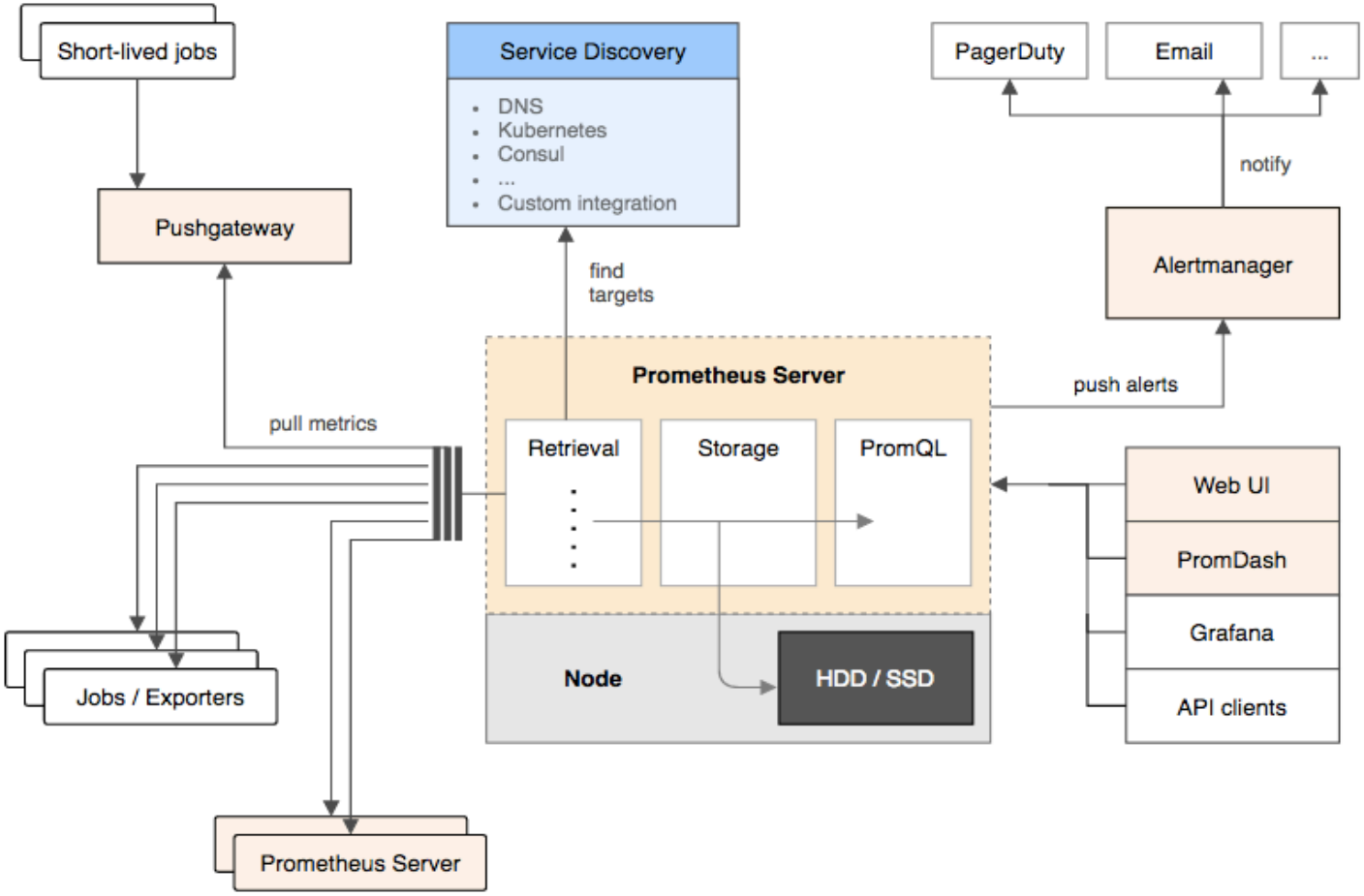
Prometheus

- Client libraries available for many languages so you can instrument your applications
- Flexible query language (think SQL for for time-series)
- Command line querying tool
- Built-in alerting
- Can be federated for aggregation of data in multiple data centers

Prometheus

- Go standalone application (no dependencies)
- All metrics stored, no loss of granularity
- Similar format to graphite, with the added dimensions:
  - freeswitch_total_calls{profile=internal} 20
    1470843323

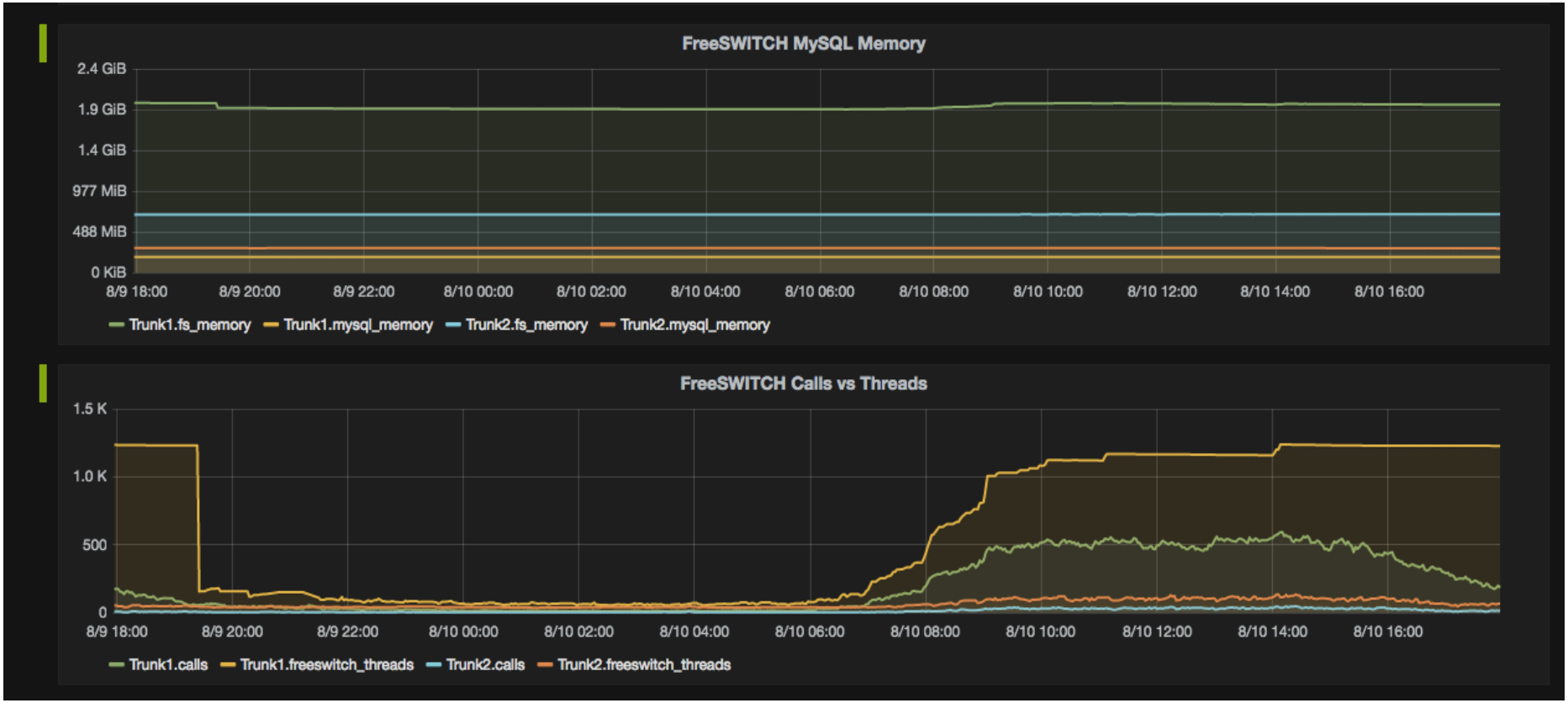
Prometheus


Short-lived jobs
Pushgateway
Service Discovery
DNS
Kubernetes
Consul
...
Custom integration
find
targets
PagerDuty
Email
...
notify
Alertmanager
Prometheus Server
Retrieval
Storage
PromQL
push alerts
pull metrics
Web UI
PromDash
Grafana
API clients
Node
HDD / SSD
Jobs / Exporters
Prometheus Server

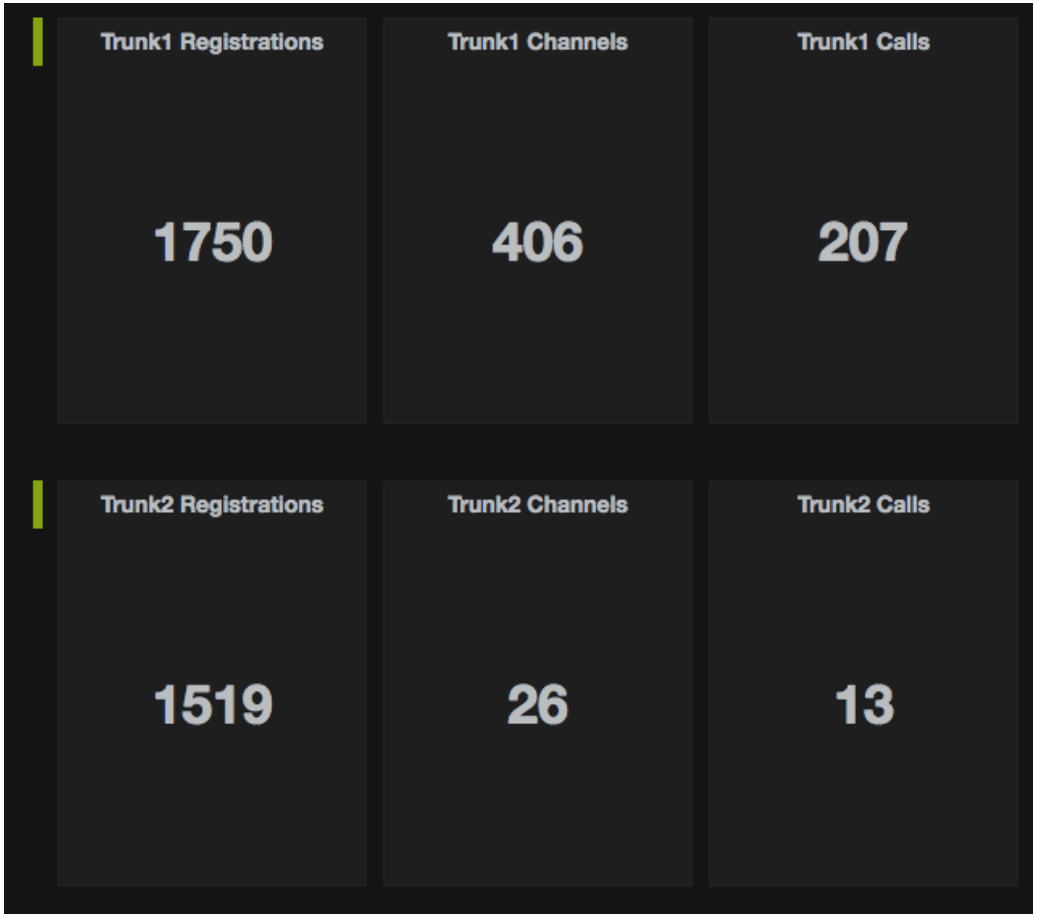
Graphing with Grafana

- Powerful dashboard builder application
- Support for many backend databases (Prometheus, Graphite)
- Automate creation of dashboards with their HTTP API and JSON dashboard definition format
- Rich graphing customizations, styling, access control, etc

Graphing with Grafana



Graphing with Grafana



mod_prometheus

- Native Prometheus FreeSWITCH integration module written in Rust
- Early development (started just a few days ago) but already provides useful metrics out of the box, ready to be scrapped by Prometheus
- Metrics like Calls, Failed calls, Registrations, CPS, ASR, etc
- You can then use Prometheus Alert manager to create alert trigger rules based on those metrics
- https://github.com/moises-silva/mod_prometheus

mod_prometheus

- Installing requires Rust and Cargo (the Rust package/build tool and FreeSWITCH master)

git clone https://github.com/moises-silva/mod_prometheus
cd mod_prometheus
cargo build
cp target/debug/libmod_prometheus.so /usr/local/freeswitch/mod

(Do not attempt to rename the module to mod_prometheus, it won't work)

mod_prometheus

- Then just load it

```
freeswitch@sigchld> load libmod_prometheus
2016-08-11 05:03:51.179789 [INFO] mod_enum.c:880 ENUM Reloaded
2016-08-11 05:03:51.179789 [INFO] switch_time.c:1415 Timezone reloaded 1750 definitions
2016-08-11 05:03:51.179789 [INFO] mod_prometheus.rs:238 Loaded Prometheus Metrics Module
2016-08-11 05:03:51.179789 [CONSOLE] switch_loadable_module.c:1538 Successfully Loaded [mod_prometheus]
2016-08-11 05:03:51.179789 [NOTICE] switch_loadable_module.c:338 Adding API Function 'prom_counter_increase'
2016-08-11 05:03:51.179789 [NOTICE] switch_loadable_module.c:338 Adding API Function 'prom_gauge_set'

+OK Reloading XML
+OK
```

mod_prometheus

- Test with curl

```
moy@sigchld ~
$ curl -v http://127.0.0.1:6780/metrics
*   Trying 127.0.0.1...
* Connected to 127.0.0.1 (127.0.0.1) port 6780 (#0)
> GET /metrics HTTP/1.1
> Host: 127.0.0.1:6780
> User-Agent: curl/7.50.0
> Accept: */*
>
< HTTP/1.1 200 OK
< Server: tiny-http (Rust)
< Date: Thu, 11 Aug 2016 09:16:10 GMT
< Content-Type: text/plain; version=0.0.4
< Content-Length: 1874
<
# HELP freeswitch_heartbeats FreeSWITCH heartbeat count
freeswitch_heartbeats 3 1470906970815
# HELP freeswitch_sessions FreeSWITCH Session Count
freeswitch_sessions 0 1470906970815
# HELP freeswitch_sessions_answered FreeSWITCH Answered Sessions Count
freeswitch_sessions_answered 0 1470906970815
# HELP freeswitch_sessions_failed FreeSWITCH Failed Sessions Count
freeswitch_sessions_failed 0 1470906970815
# HELP freeswitch_sessions_inbound FreeSWITCH Inbound Sessions Count
freeswitch_sessions_inbound 0 1470906970815
# HELP freeswitch_sessions_inbound_answered FreeSWITCH Answered Inbound Sessions Count
freeswitch_sessions_inbound_answered 0 1470906970815
```

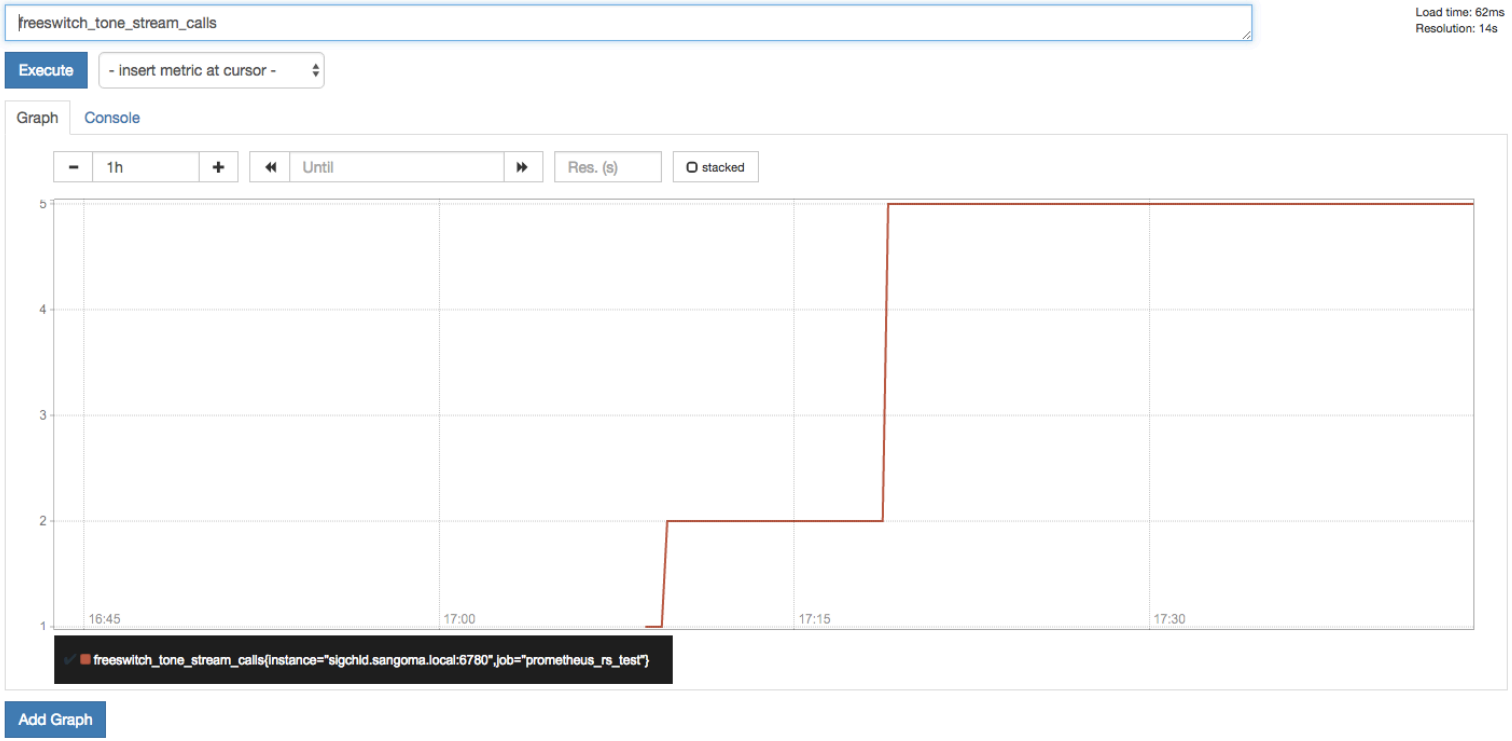
mod_prometheus

- Use custom counters and gauges from the dialplan

```
<extension name="tone_stream">
  <condition field="destination_number" expression="^9198$">
    <action application="set" data="api_result=${prom_counter_increase(freeswitch_tone_stream_calls)}"/>
    <action application="answer"/>
    <action application="playback" data="{loops=10}tone_stream://path=${conf_dir}/tetris.ttml"/>
  </condition>
</extension>
```

mod_prometheus

- Next Prometheus scrap will get those metrics



Final Thoughts

- Application instrumentation must be part of application development
- Monitoring tools of all kinds are available. No excuses for lack of monitoring
- Monitor all the things!

Afterword

- Bringing Rust to FreeSWITCH. The mod_prometheus module is the first Rust module coming into FreeSWITCH
- Rust is a new systems programming language within the same speed range as C/C++ but with memory safety guarantees using an ownership system and move semantics by default
- Rust bindings are available by the freeswitchrs project created by Michael Giagnocavo: https://gitlab.com/wiresight/freeswitchrs/
- Rust memory safety could make for an interesting addition to the stability of FreeSWITCH. If you're writing modules please consider using Rust

QUESTIONS

Contact Us

- Sangoma Technologies

  100 Renfrew Drive, Suite 100
  Markham, Ontario L3R 9R6
  Canada

- Website

  http://www.sangoma.com/

- Telephone

  +1 905 474 1990 x2 (for Sales)

- Email

  sales@sangoma.com

/Sangoma

/Sangoma

/SangomaTechnologies

blog.sangoma.com

THANK YOU

SANGOMA

CONNECT WITH SANGOMA