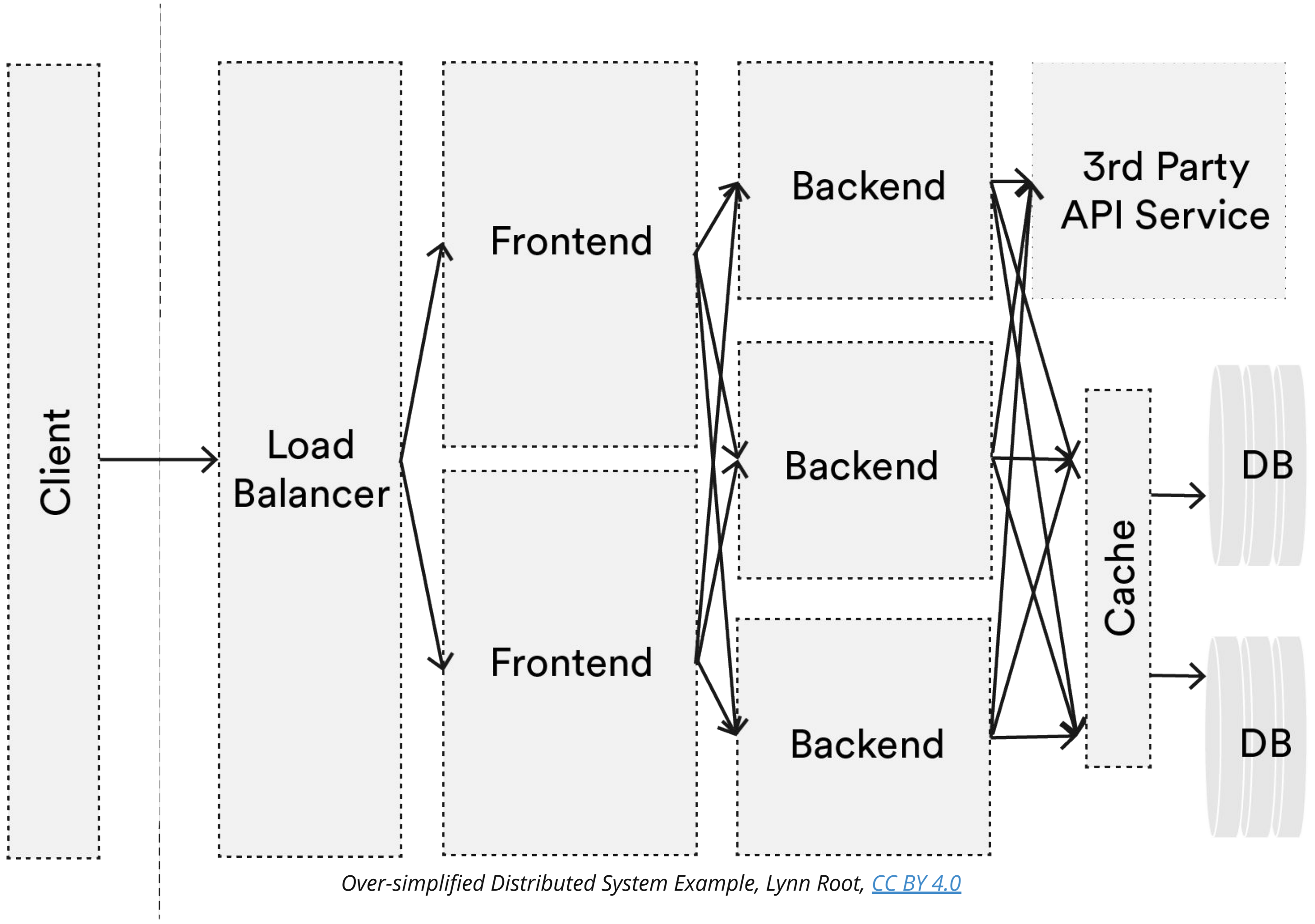
Adding the three pillars of Observability to your Python app

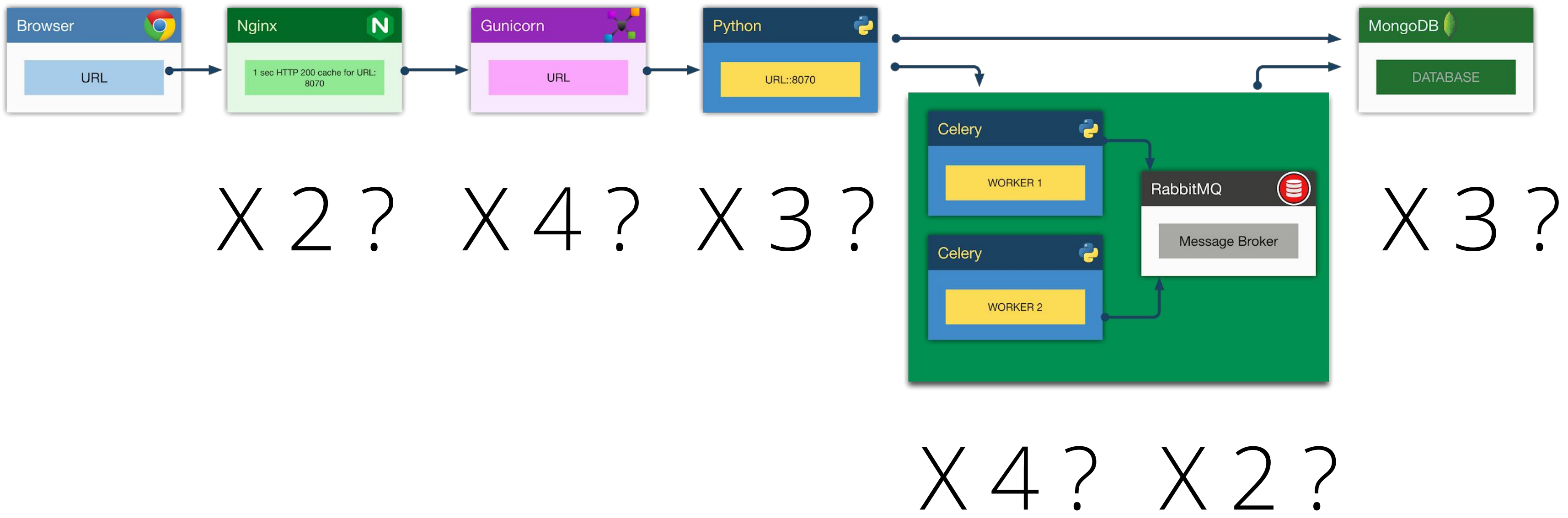
Eoin Brazil, PhD, MSc, Team Lead, MongoDB

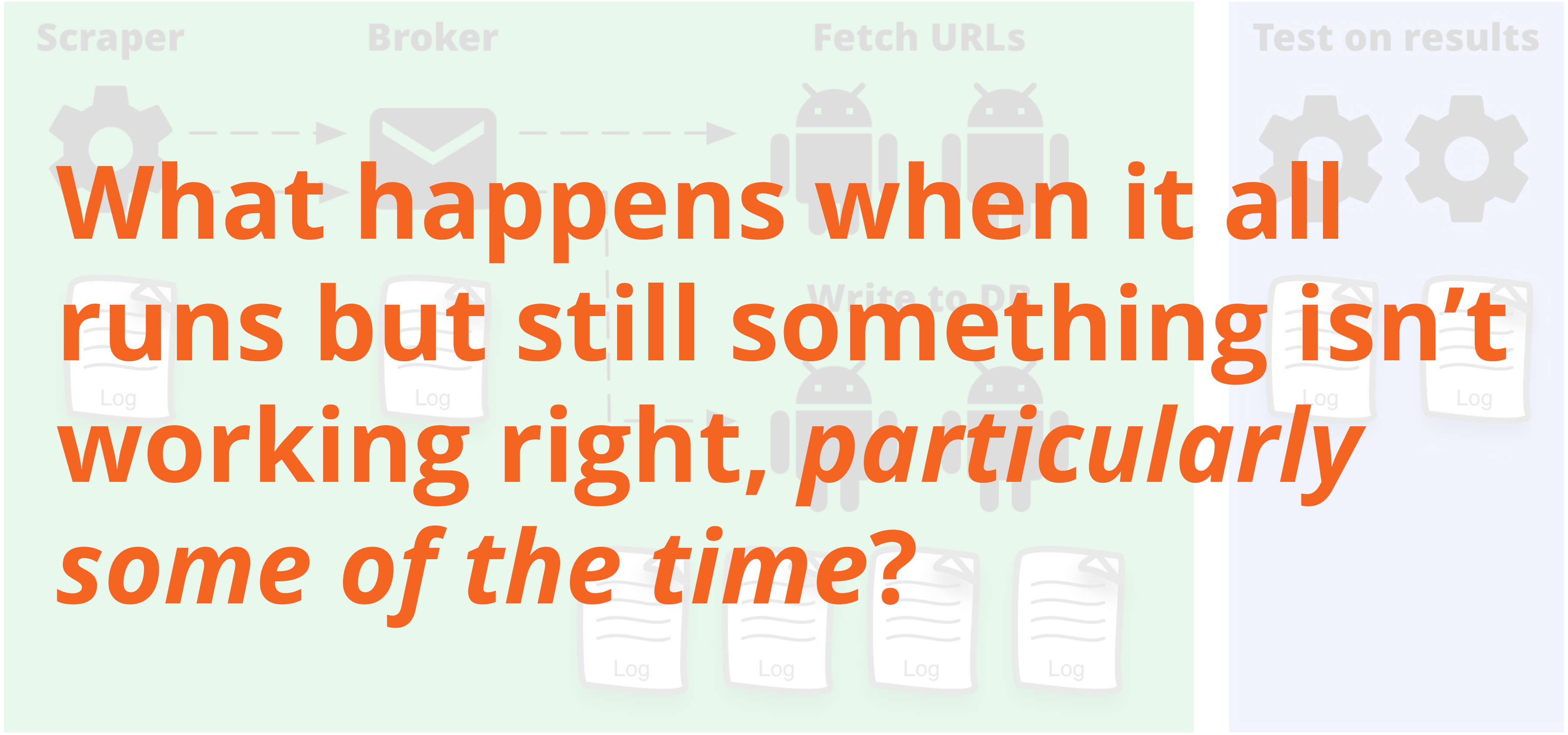


Over-simplified Distributed System Example, Lynn Root, CC BY 4.0

Tracing, Fast and Slow by Lynn Root

Distributed Systems or Your Standard Web Stack ?





What happens when it all runs but still something isn't working right, *particularly some of the time?*

Observability

Make complex systems transparent to enable understanding of the systems state.

Pillars - **Logs** & **Metrics** & **Events**

Monitoring

Aims to report the overall health of systems.

Strong overlap with aspects of **Metrics** but focus for Application side for this talk.

Observability vs Monitoring

Whitebox	Blackbox
Metrics	Monitoring
Logs	Polling
Traces	Uptime

Monitoring - Patterns

- Utilisation, Saturation, Errors (USE)
- For each resource, Rate (RPS), Errors, Duration (RED method)
- Golden Signals (Latency, Errors, Traffic, Saturation)

Observability vs Monitoring

Enable understanding with context, ideal for debugging. *Unknown* failure modes.

Snapshot of overall health of systems. *Known* failure modes.

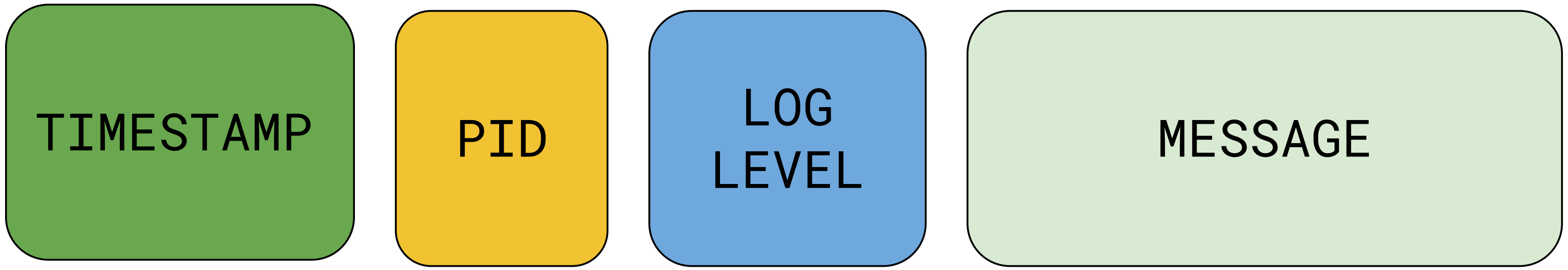
Logs

Logs

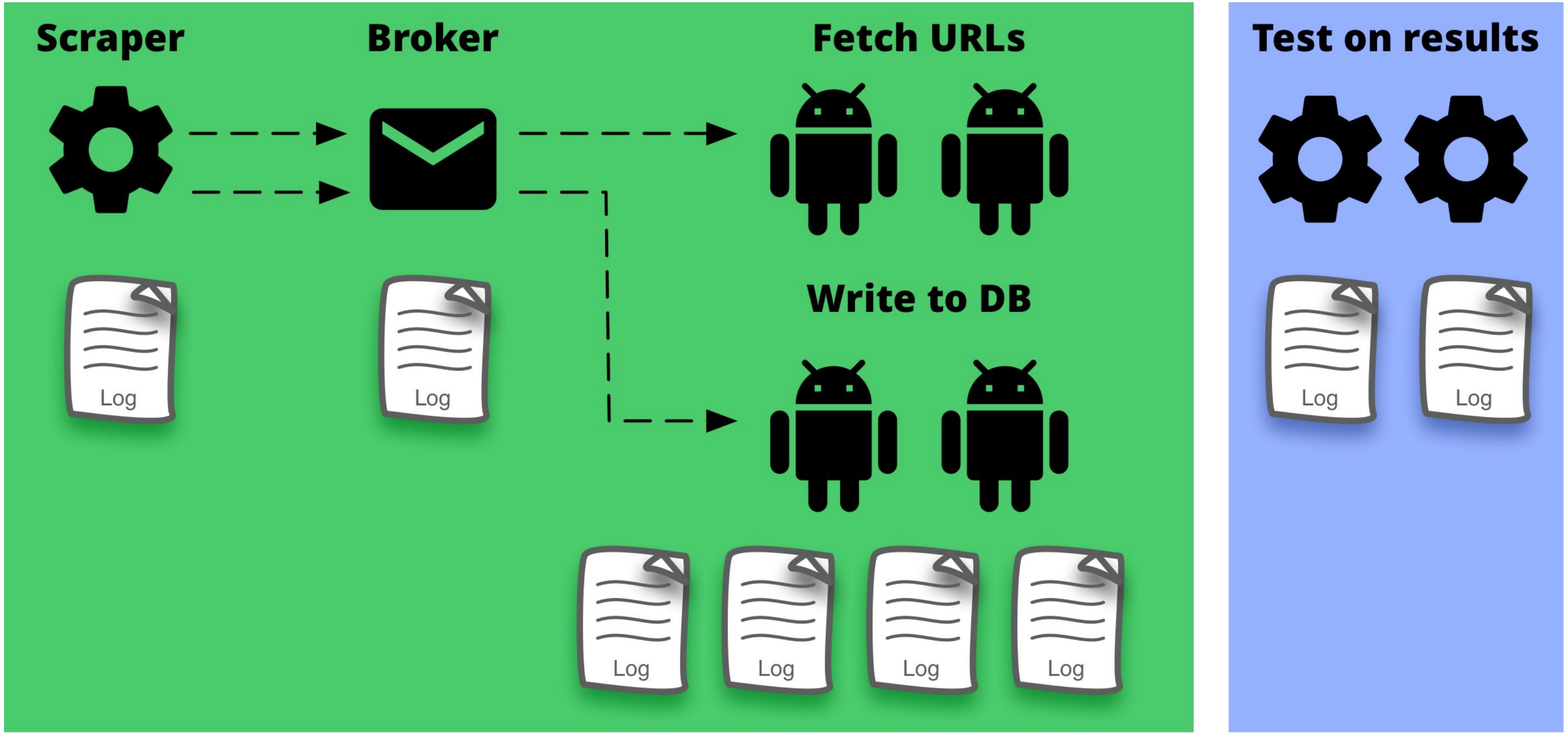
- Typically, loosely structured requests, errors, or other messages in a sequence of rotating text files.
- Can be structured and should be.
- Specialised additions - exception trackers (Sentry, Rollbar, etc.)

Logs - Semi Structured

```
[2018-10-17 20:00:17 +0100] [33353] [INFO] Goin' Fast @ http://0.0.0.0:8006
[2018-10-17 20:00:17 +0100] [33353] [INFO] Starting worker [33353]
[2018-10-17 20:18:20 +0100] - (sanic.access)[INFO][127.0.0.1:59076]: GET
http://127.0.0.1:8006/  200 829
```



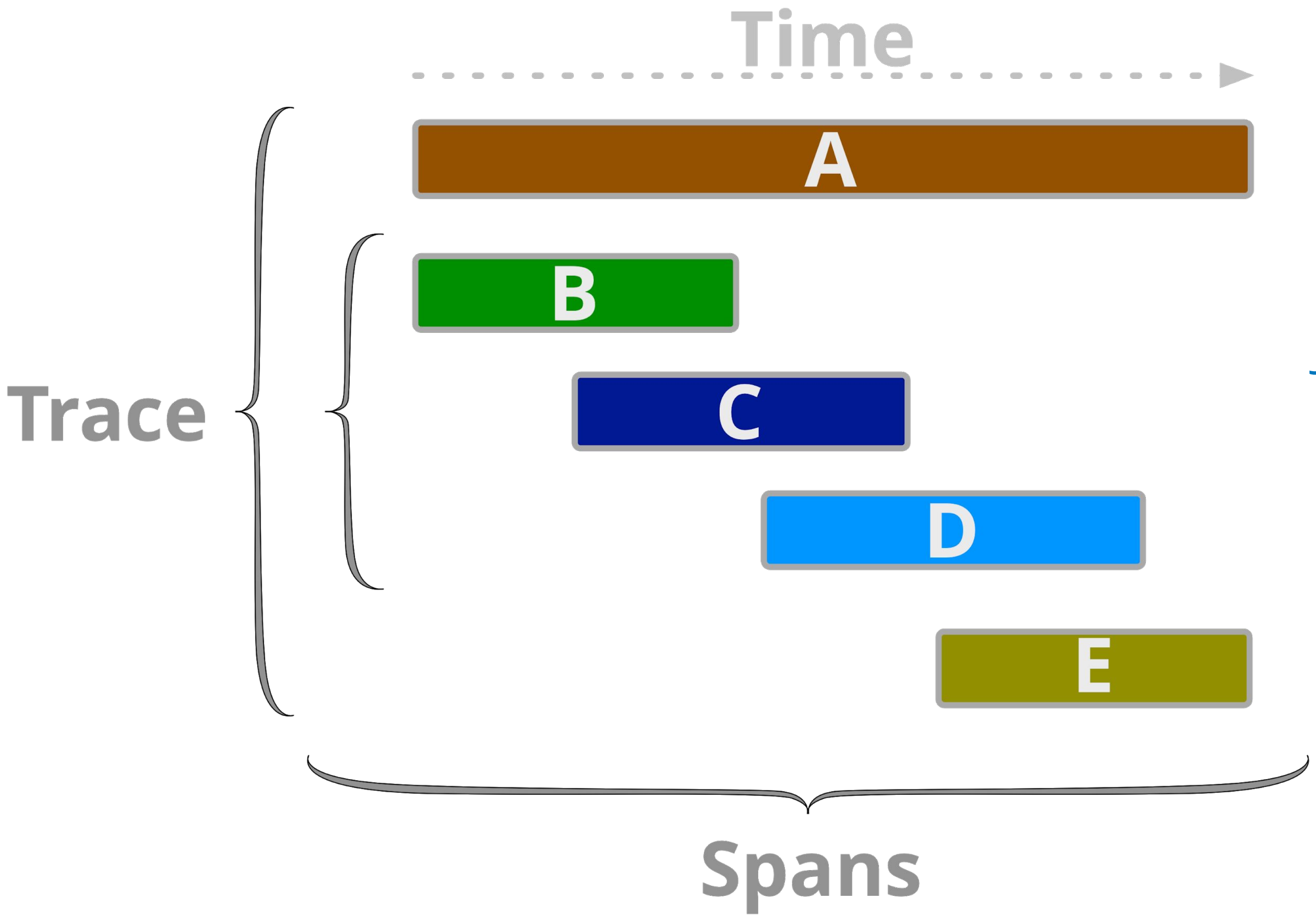
My own software problems/learnings



Logs - 3 Steps to add structure

- Add UUIDs to requests (spans)
- Use key-value pairs instead of text
- Use JSON instead of plain text

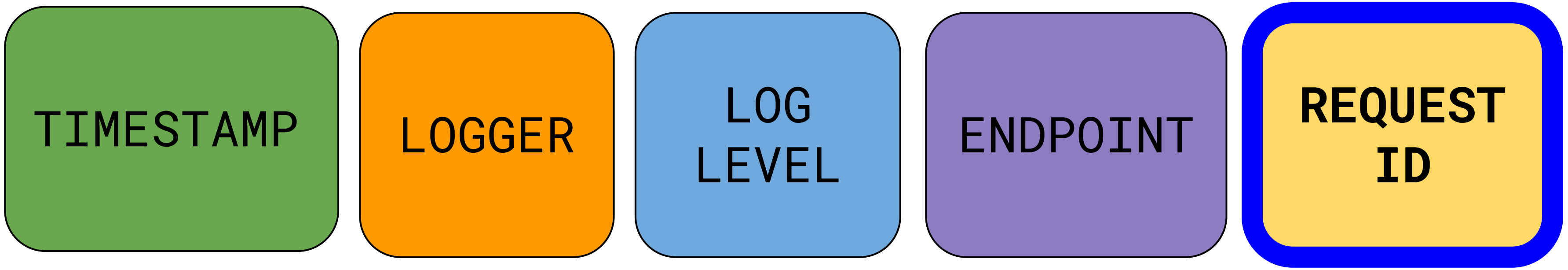
Structlog & UUID



[Jaegar Tracing Architecture]

Logs - UUID

```
2018-10-24 14:01:47,331 - 89195 - INFO - main - {
 "endpoint": "/",
 "level": "info",
 "logger": "__main__",
 "request_id": "UUID('6fafaa91-eca0-4d4a-a9f8-0c441a01790b')",
 "timestamp": "2018-10-24T13:01:47.330811Z"
}
```



Logs - 3 Steps to add structure

- Add UUIDs to requests (spans)
- Use key-value pairs instead of text
- Use JSON instead of plain text

Structlog & UUID

Metrics

Metrics

Application metrics, *statsd* was the forerunner of many of this category.

- *How many requests made ? How many failures ? What types of failures ? Service checks ?*

Metrics - statsd

```
>>> import statsd
>>> c = statsd.StatsClient('localhost', 8125)
>>> c.incr('auth.success')
>>> c.timing('login.timer', 320)
```

Metrics - DogStatsD

```
>>> from datadog import statsd
>>> from datadog.api.constants import CheckStatus
>>> statsd.increment('index.response.total',
tags=['code=200'])
>>> statsd.event('deploy','app: pycon.ie\n' +
'version: ' + githash + 'env: live')
>>> statsd.service_check(check_name='pycon',
status='Checkstatus.OK', message='Response: 200 OK')
```

Metrics - Prometheus

Time series metric name with KV pairs (labels)

- UDP packet every time a metric is recorded (statsd) vs aggregate in-process and submit them every few seconds (Prometheus)

Logs and Metrics overlap

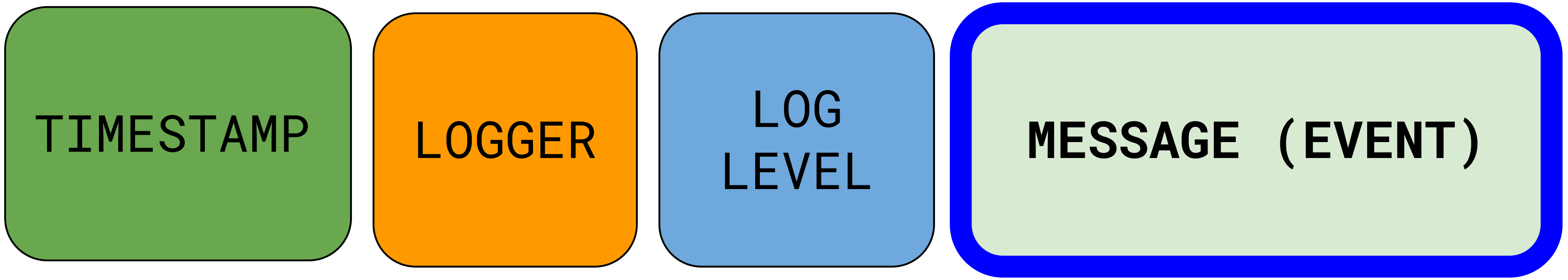
Metrics are a snapshot with counters and gauges (short period).

Log derived metrics, granular info, holistic view more easily aggregated.

Events

Logs - Structured (structlog)

```
2018-10-24 13:51:02,136 - 89028 - INFO - main - {
 "event": "Start running API",
 "level": "info",
 "logger": "__main__",
 "timestamp": "2018-10-24T12:51:02.136399Z"
}
```



Why Structured Logs & JSON ?

Remains human readable

Makes it easier to specific event via associated data

JSON simplifies log aggregator’s job

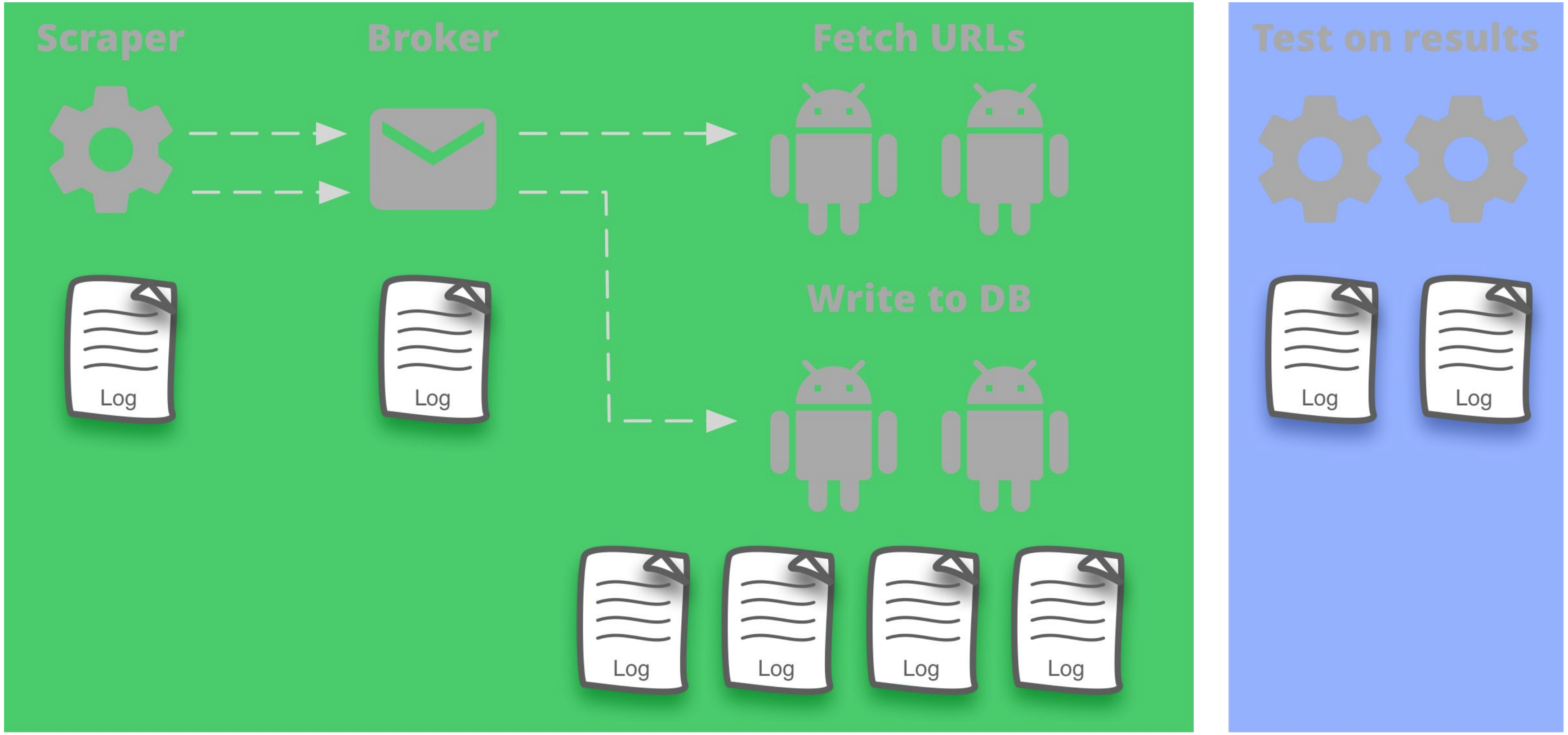
Log Aggregators

Graylog, ELK, Splunk, FluentD, etc ....

A key is a group-by target allows for new types of questions to be asked easily.

Issue/Incident remediation & historic trends (business intelligence)

My own software problems/learnings

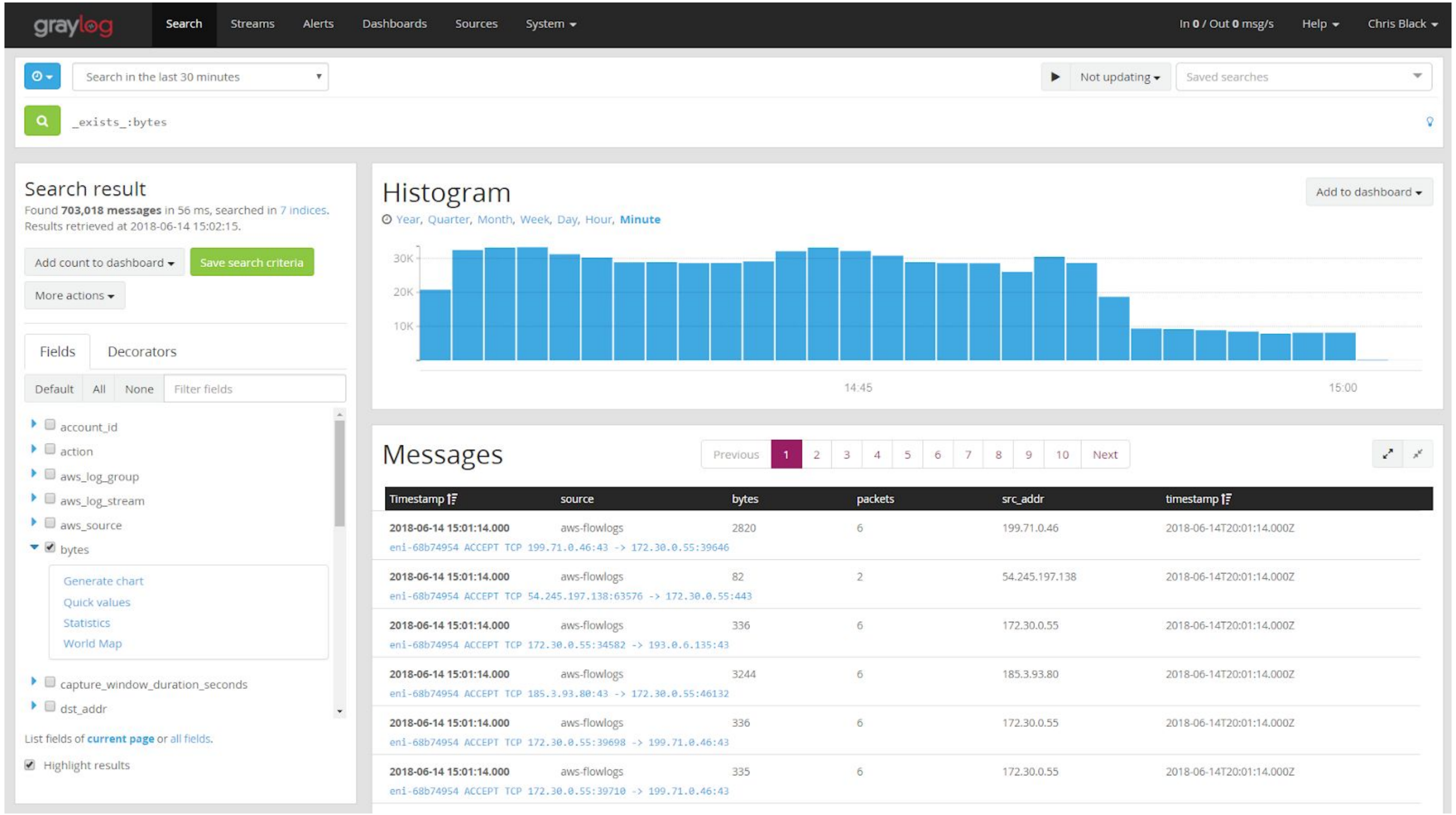


Graylog

1) Aggregates and extracts important data from server logs, which are often sent using the Syslog protocol.

2) It also allows you to search and visualize the logs in a web interface.

Graylog - Query bytes exist



```
_exists_:bytes
```

Search result

Found **703,018 messages** in 56 ms, searched in 7 indices.
Results retrieved at 2018-06-14 15:02:15.

List fields of **current page** or all fields.

☑ Highlight results

Messages

Timestamp	source	bytes	packets	src_addr	timestamp
2018-06-14 15:01:14.000	aws-flowlogs	2820	6	199.71.0.46	2018-06-14T20:01:14.000Z
`eni-68b74954 ACCEPT TCP 199.71.0.46:43 -> 172.30.0.55:39646`					
2018-06-14 15:01:14.000	aws-flowlogs	82	2	54.245.197.138	2018-06-14T20:01:14.000Z
`eni-68b74954 ACCEPT TCP 54.245.197.138:63576 -> 172.30.0.55:443`					
2018-06-14 15:01:14.000	aws-flowlogs	336	6	172.30.0.55	2018-06-14T20:01:14.000Z
`eni-68b74954 ACCEPT TCP 172.30.0.55:34582 -> 193.0.6.135:43`					
2018-06-14 15:01:14.000	aws-flowlogs	3244	6	185.3.93.80	2018-06-14T20:01:14.000Z
`eni-68b74954 ACCEPT TCP 185.3.93.80:43 -> 172.30.0.55:46132`					
2018-06-14 15:01:14.000	aws-flowlogs	336	6	172.30.0.55	2018-06-14T20:01:14.000Z
`eni-68b74954 ACCEPT TCP 172.30.0.55:39698 -> 199.71.0.46:43`					
2018-06-14 15:01:14.000	aws-flowlogs	335	6	172.30.0.55	2018-06-14T20:01:14.000Z
`eni-68b74954 ACCEPT TCP 172.30.0.55:39710 -> 199.71.0.46:43`					

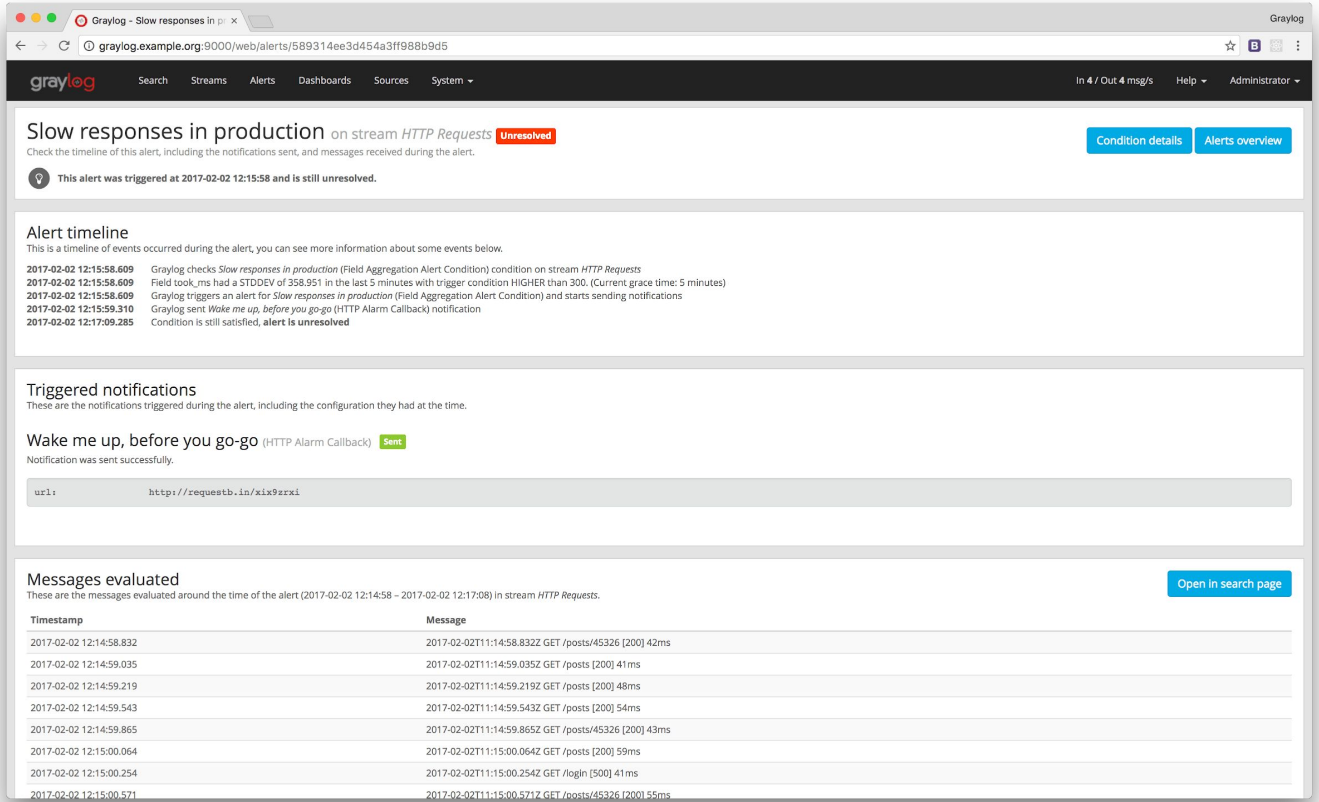
Source: https://www.graylog.org/post/trend-analysis-with-graylog

Beyond a Browser UI to Logs ?

Show the number of calls for all API methods by name?

Log your API methods by name
Tags allow you to use group-by

Graylog - Alerting



Slow responses in production *on stream HTTP Requests* Unresolved

Check the timeline of this alert, including the notifications sent, and messages received during the alert.

This alert was triggered at 2017-02-02 12:15:58 and is still unresolved.

Alert timeline

This is a timeline of events occurred during the alert, you can see more information about some events below.

2017-02-02 12:15:58.609 Graylog checks *Slow responses in production* (Field Aggregation Alert Condition) condition on stream *HTTP Requests*
2017-02-02 12:15:58.609 Field took_ms had a STDDEV of 358.951 in the last 5 minutes with trigger condition HIGHER than 300. (Current grace time: 5 minutes)
2017-02-02 12:15:58.609 Graylog triggers an alert for *Slow responses in production* (Field Aggregation Alert Condition) and starts sending notifications
2017-02-02 12:15:59.310 Graylog sent *Wake me up, before you go-go* (HTTP Alarm Callback) notification
2017-02-02 12:17:09.285 Condition is still satisfied, **alert is unresolved**

Triggered notifications

These are the notifications triggered during the alert, including the configuration they had at the time.

Wake me up, before you go-go (HTTP Alarm Callback) Sent

Notification was sent successfully.

```
url:            http://requestb.in/xix9zrxi
```

Messages evaluated

These are the messages evaluated around the time of the alert (2017-02-02 12:14:58 – 2017-02-02 12:17:08) in stream *HTTP Requests*.

Timestamp	Message
2017-02-02 12:14:58.832	2017-02-02T11:14:58.832Z GET /posts/45326 [200] 42ms
2017-02-02 12:14:59.035	2017-02-02T11:14:59.035Z GET /posts [200] 41ms
2017-02-02 12:14:59.219	2017-02-02T11:14:59.219Z GET /posts [200] 48ms
2017-02-02 12:14:59.543	2017-02-02T11:14:59.543Z GET /posts [200] 54ms
2017-02-02 12:14:59.865	2017-02-02T11:14:59.865Z GET /posts/45326 [200] 43ms
2017-02-02 12:15:00.064	2017-02-02T11:15:00.064Z GET /posts [200] 59ms
2017-02-02 12:15:00.254	2017-02-02T11:15:00.254Z GET /login [500] 41ms
2017-02-02 12:15:00.571	2017-02-02T11:15:00.571Z GET /posts/45326 [200] 55ms

Source: http://docs.graylog.org/en/2.4/pages/streams/alerts.html

Find more on logs

- “Structured logging in Python” and “Logging as a First Class Citizen” by Steve Tarver
- http://www.structlog.org/en/stable/
- “I Heart Logs: Event Data, Stream Processing, and Data Integration” by Jay Kreps

Find more on metrics

- Measure Anything, Measure Everything (Etsy)
- Collecting Metrics Using StatsD, a Standard for Real-Time Monitoring
- Monitoring Applications with StatsD
- Logs and Metrics by Cindy Sridharan
    - https://github.com/google/mtail

Find more on events

- Tracing, Fast and Slow by Lynn Root
- Monitoring and Observability by Cindy Sridharan

Observability

Logs - UUIDs, KV pairs, Structlog, JSON, mtail
Metrics - statsd, dogstatsd
Events - Graylog, Splunk, ELK

Only the tip of the iceberg... and you still need to monitor!

Scraper
Broker
Fetch URLs
Test on results
Write to DB
Log
Log
Log
Log
Log
Log
Log
Log

~~What happens when it all runs but still something isn’t working right, *particularly some of the time?*~~

Questions ?