AUTOMATED INCIDENT RESPONSE WITH OSQUERY AND LOKI

GEORGE ADAMS IV & ED WELCH

USE CASE

DETECTING AND ALERTING SSH CONNECTIONS.

This is the end goal we'll build towards as we learn a little more along the way.

FOLLOW ALONG

HTTPS://GITHUB.COM/GEOWA4/LEARN-LOKI

OSQUERY

RELEASED BY FACEBOOK IN 2014

```
commit 73a32b729403b2f5a7c204b0f7cfb86fdfdd0a85
Author: mike@arpaia.co <mike@arpaia.co>
Date:   Wed Jul 30 17:35:19 2014 -0700

    Initial commit
```

OSQUERY

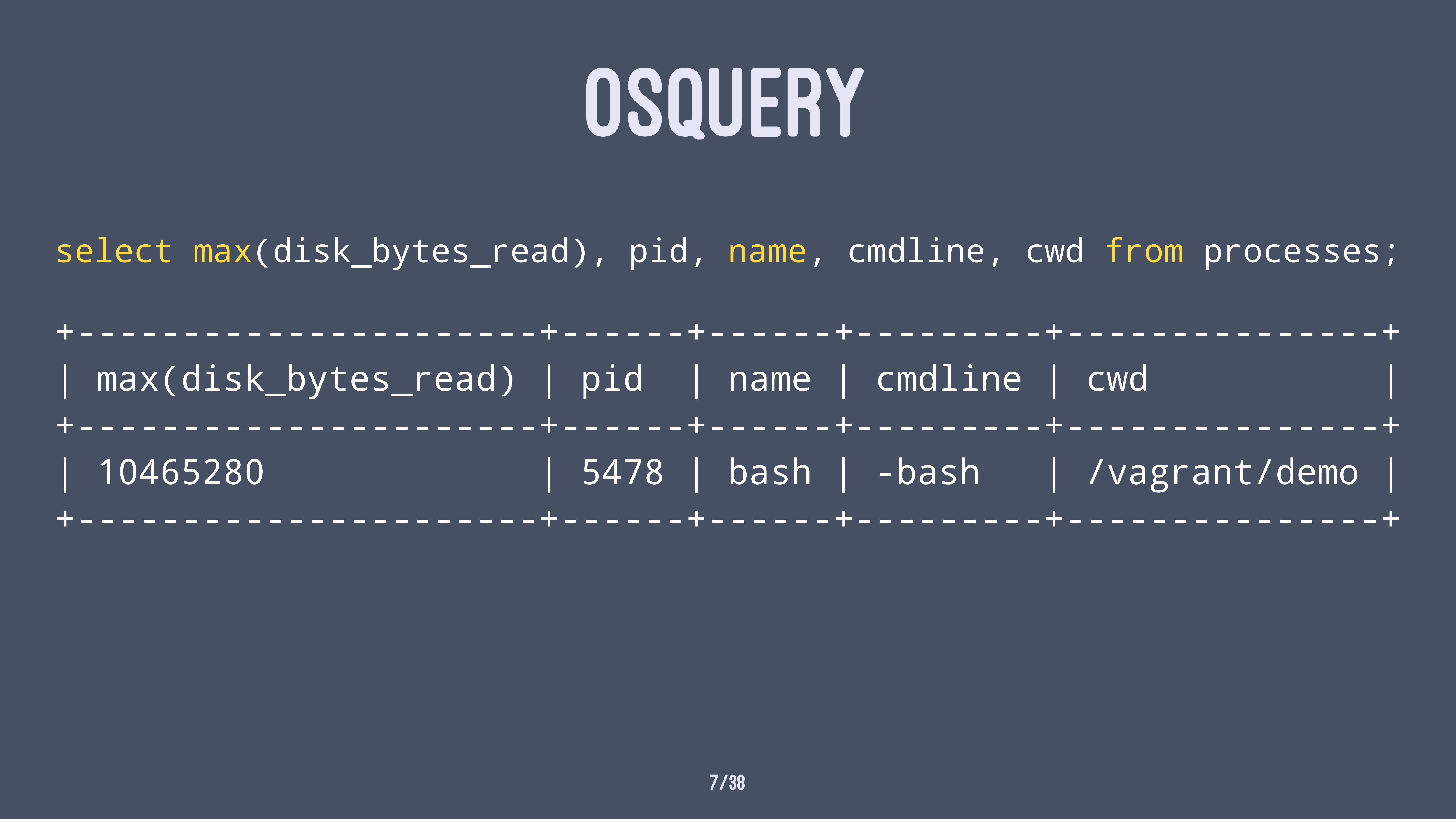
WORKS ON MY MACHINE

OSQUERY

SQL INTERFACE TO YOUR ENDPOINTS

```
.schema processes

CREATE TABLE processes(
  `pid` BIGINT, `name` TEXT, `path` TEXT, `cmdline` TEXT,
  `disk_bytes_read` BIGINT, `disk_bytes_written` BIGINT,
  ...
  PRIMARY KEY (`pid`)
) WITHOUT ROWID;
```



OSQUERY

```
select max(disk_bytes_read), pid, name, cmdline, cwd from processes;

+-----------------------+------+------+---------+----------------+
| max(disk_bytes_read) | pid  | name | cmdline | cwd            |
+-----------------------+------+------+---------+----------------+
| 10465280              | 5478 | bash | -bash   | /vagrant/demo |
+-----------------------+------+------+---------+----------------+
```

Who remembers how the command to list what processes are listening on which ports?

OSQUERY

```
select p.name, l.port, l.protocol
from processes p inner join listening_ports l on p.pid = l.pid
where p.name = 'VBoxHeadless' and l.port <> 0;

+--------------+------+----------+
| name         | port | protocol |
+--------------+------+----------+
| VBoxHeadless | 3000 | 6        |
| VBoxHeadless | 3100 | 6        |
| VBoxHeadless | 9080 | 6        |
| VBoxHeadless | 9090 | 6        |
| VBoxHeadless | 2222 | 6        |
+--------------+------+----------+
```

OSQUERY

```
select c.name, p.port, p.host_port
from docker_containers c inner join docker_container_ports p
on c.id = p.id;

+---------------------+------+-----------+
| name                | port | host_port |
+---------------------+------+-----------+
| /demo_loki_1        | 80   | 0         |
| /demo_loki_1        | 3100 | 3100      |
| /demo_grafana_1     | 3000 | 3000      |
| /demo_prometheus_1  | 9090 | 9090      |
| /demo_promtail_1    | 9080 | 9080      |
+---------------------+------+-----------+
```

OSQUERY

```
.schema last
```

```
CREATE TABLE last(
  `username` TEXT, `time` INTEGER, `host` TEXT,
  `pid` INTEGER, `tty` TEXT, `type` INTEGER
);
```

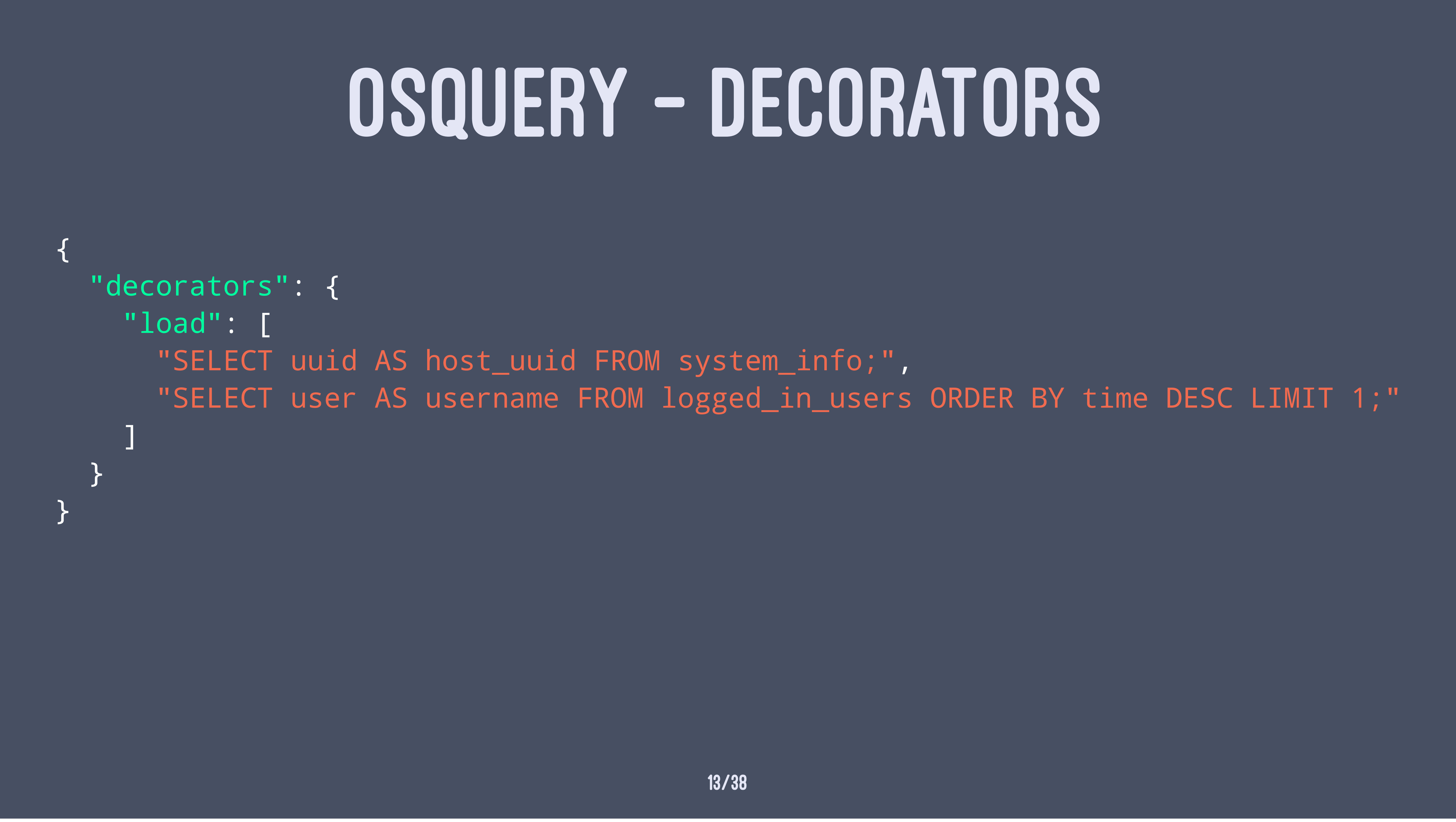
OSQUERY

```
select * from last;
```

```
+----------+-------+------+------+------------+-------------------+
| username | tty   | pid  | type | time       | host              |
+----------+-------+------+------+------------+-------------------+
| reboot   | ~     | 0    | 2    | 1566866107 | 4.15.0-52-generic |
| runlevel | ~     | 53   | 1    | 1566866118 | 4.15.0-52-generic |
|          | ttyS0 | 859  | 5    | 1566866119 |                   |
| LOGIN    | ttyS0 | 859  | 6    | 1566866119 |                   |
|          | tty1  | 879  | 5    | 1566866119 |                   |
| LOGIN    | tty1  | 879  | 6    | 1566866119 |                   |
| vagrant  | pts/0 | 5396 | 7    | 1566869034 | 10.0.2.2          |
| vagrant  | pts/1 | 6465 | 7    | 1566870878 | 10.0.2.2          |
|          | pts/1 | 6465 | 8    | 1566870880 |                   |
| vagrant  | pts/1 | 6571 | 7    | 1566870886 | 10.0.2.2          |
|          | pts/1 | 6571 | 8    | 1566870910 |                   |
+----------+-------+------+------+------------+-------------------+
```

OSQUERY - PACKS

```
{
  "queries": {
    "last": {
      "query": "select * from last;",
      "interval": "60",
      "platform": "posix",
      "version": "1.4.5",
      "description": "..."
    }
  }
}
```



OSQUERY - DECORATORS

```
{
  "decorators": {
    "load": [
      "SELECT uuid AS host_uuid FROM system_info;",
      "SELECT user AS username FROM logged_in_users ORDER BY time DESC LIMIT 1;"
    ]
  }
}
```

Decorator results are added to pack results.
If you're on AWS, you can add a query for your EC2's tags.

OSQUERY - RESULTS

/var/log/osquery/osqueryd.results.log

```
{
  "name": "pack_incident-response_last",
  "hostIdentifier": "ubuntu-bionic",
  "calendarTime": "Tue Aug 27 01:55:13 2019 UTC",
  "decorations": {
    "host_uuid": "2401CCE9-23EA-4D4D-8C84-D5C8437EBE15",
    "username": "vagrant"
  },
  "columns": {
    "host": "10.0.2.2",
    "pid": "6465",
    "time": "1566870878",
    "tty": "pts/1",
    "type": "7",
    "username": "vagrant"
  },
  "action": "added"
}
```

OSQUERY - RECAP

IT'S BEEN AROUND A WHILE

CROSS-PLATFORM

IT'S JUST SQL

SCHEDULE QUERIES WITH "PACKS"

LOKI

CLOUD-NATIVE LOG AGGREGATION

MADE BY GRAFANA

LOKI

PROMETHEUS-INSPIRED LOGGING FOR CLOUD NATIVES.

To know what this means, let's take a quick detour to Prometheus.

PROMETHEUS

METRICS COLLECTION VIA "PULL"

TIME-SERIES DATA STORE

QUERYABLE VIA PROMQL

PROMETHEUS - PUSH VS. PULL

TODO: diagram push vs pull

PROMETHEUS - SCRAPING

```
scrape_configs:
  - job_name: "promtail"
    static_configs:
      - targets:
          - promtail:9080
```

PROMETHEUS - DATA STRUCTURE

METRICS HAVE LABELS IN ADDITION TO VALUES.

```
rss_enjoyment{track="tech", talk="osquery_loki"} 11
```

LOKI - DATA STRUCTURE

LOG ENTRIES HAVE LABELS, TOO.

```
{track="tech", talk="osquery_loki"} "best talk ever"
{track="tech", talk="osquery_loki"} "i want to know more"
{track="tech", talk="osquery_loki"} "i hear that one guy runs rocdev"
```

LOKI - LABELS

THE MATCHING LABELS ALLOW US TO SWITCH BACK AND FORTH FREELY.

LOKI - QUERY

LOGQL

```
$ logcli query --tail '{name="pack_incident-response_last"}'
2019-08-29T03:01:37Z
{filename="/var/log/osquery/osqueryd.results.log", job="osquery_results", name="pack_incident-response_last"}
{
  "name":"pack_incident-response_last",
  "hostIdentifier":"ubuntu-bionic",
  "calendarTime":"Thu Aug 29 03:01:37 2019 UTC",
  "unixTime":1567047697,
  "epoch":0,
  "counter":115,
  "decorations":{"host_uuid":"661449FD-E11A-462B-9EA9-63A3EE8F9BDC","username":"vagrant"},
  "columns":{"host":"10.0.2.2","pid":"7404","time":"1567047680","tty":"pts/1","type":"7","username":"vagrant"},
  "action":"added"
}
```

This is why Loki is like Prometheus. We read from them the same way with the same labels.
But there's one more link with Prometheus and Loki: metrics extraction.

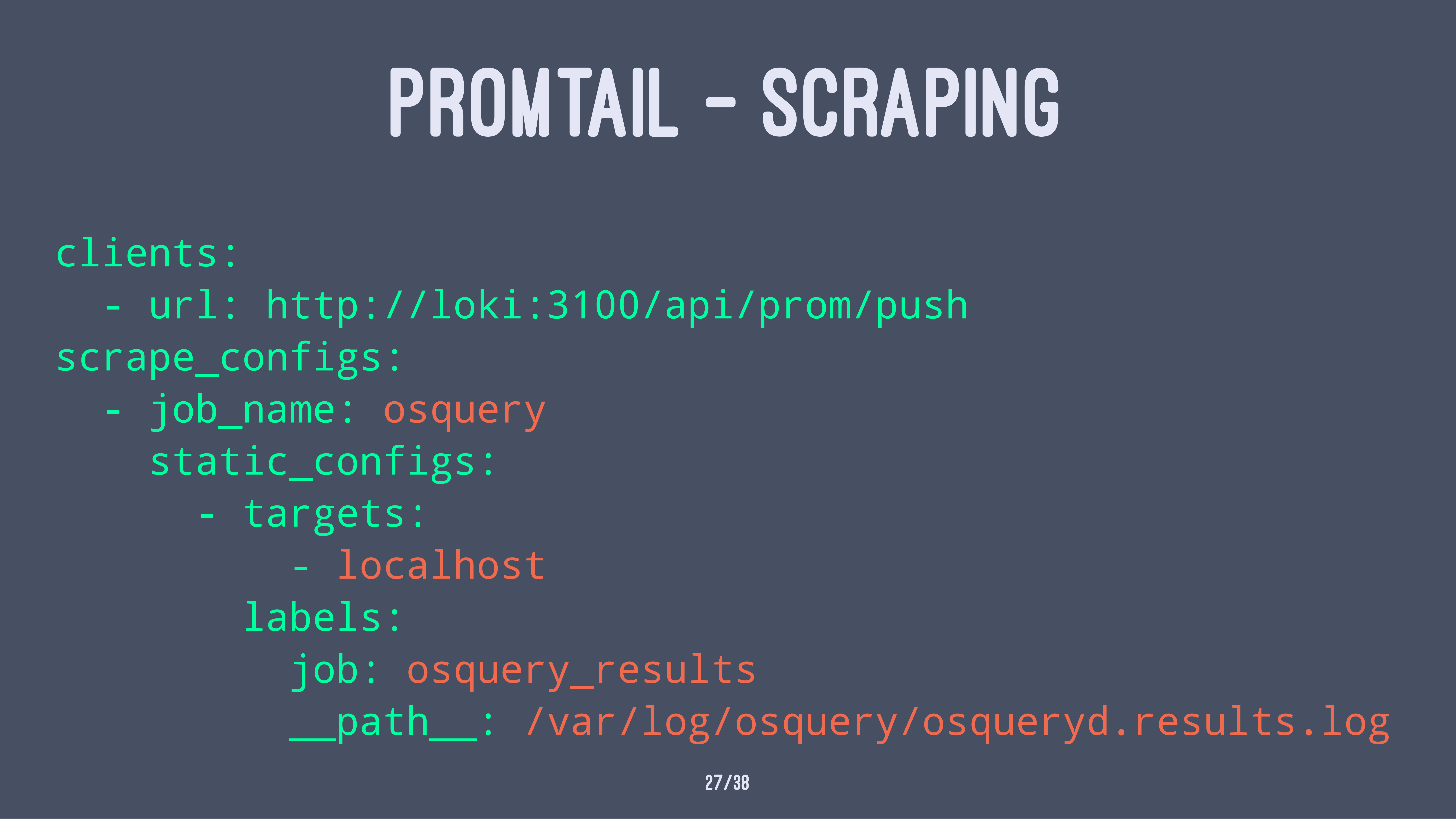
LOKI - COLLECTION

PROMTAIL

We've seen how Osquery generates logs.
We've seen how to read logs.
But how did they get there?

PROMTAIL

FORWARDS LOGS
AND
EXTRACTS METRICS



PROMTAIL - SCRAPING

```
clients:
  - url: http://loki:3100/api/prom/push
scrape_configs:
  - job_name: osquery
    static_configs:
      - targets:
          - localhost
        labels:
          job: osquery_results
          __path__: /var/log/osquery/osqueryd.results.log
```

Anything Osquery reports will be forwarded to Loki

PROMTAIL - RESULT REMINDER

```
{
  "name": "pack_incident-response_last",
  "hostIdentifier": "ubuntu-bionic",
  "calendarTime": "Thu Aug 29 03:01:37 2019 UTC",
  "unixTime": 1567047697,
  "epoch": 0,
  "counter": 115,
  "decorations": {
    "host_uuid": "661449FD-E11A-462B-9EA9-63A3EE8F9BDC",
    "username": "vagrant"
  },
  "columns": {
    "host": "10.0.2.2",
    "pid": "7404",
    "time": "1567047680",
    "tty": "pts/1",
    "type": "7",
    "username": "vagrant"
  },
  "action": "added"
}
```

PROMTAIL - PIPELINES

```
pipeline_stages:
  - json:
      expressions:
        timestamp: unixTime
        ip: columns.host
        name: name
        username: columns.username
  - timestamp:
      source: timestamp
      format: Unix
  - labels:
      name: name
```

PROMTAIL - METRICS

```
pipeline_stages:
  - ...
  - metrics:
      last_logins:
        type: Counter
        description: count last logins
        source: name
        config:
          value: pack_incident-response_last
          action: inc
```

PROMTAIL - PROMETHEUS

```yaml
scrape_configs:
  - job_name: "promtail"
    static_configs:
      - targets:
          - promtail:9080
```

WHERE WE ARE NOW

OSQUERY PRODUCING RESULTS

PROMTAIL FORWARDING TO LOKI

QUERY AND TAIL LOGS IN LOKI

PROMTAIL EXTRACTING METRICS

PROMETHEUS SCRAPING PROMTAIL

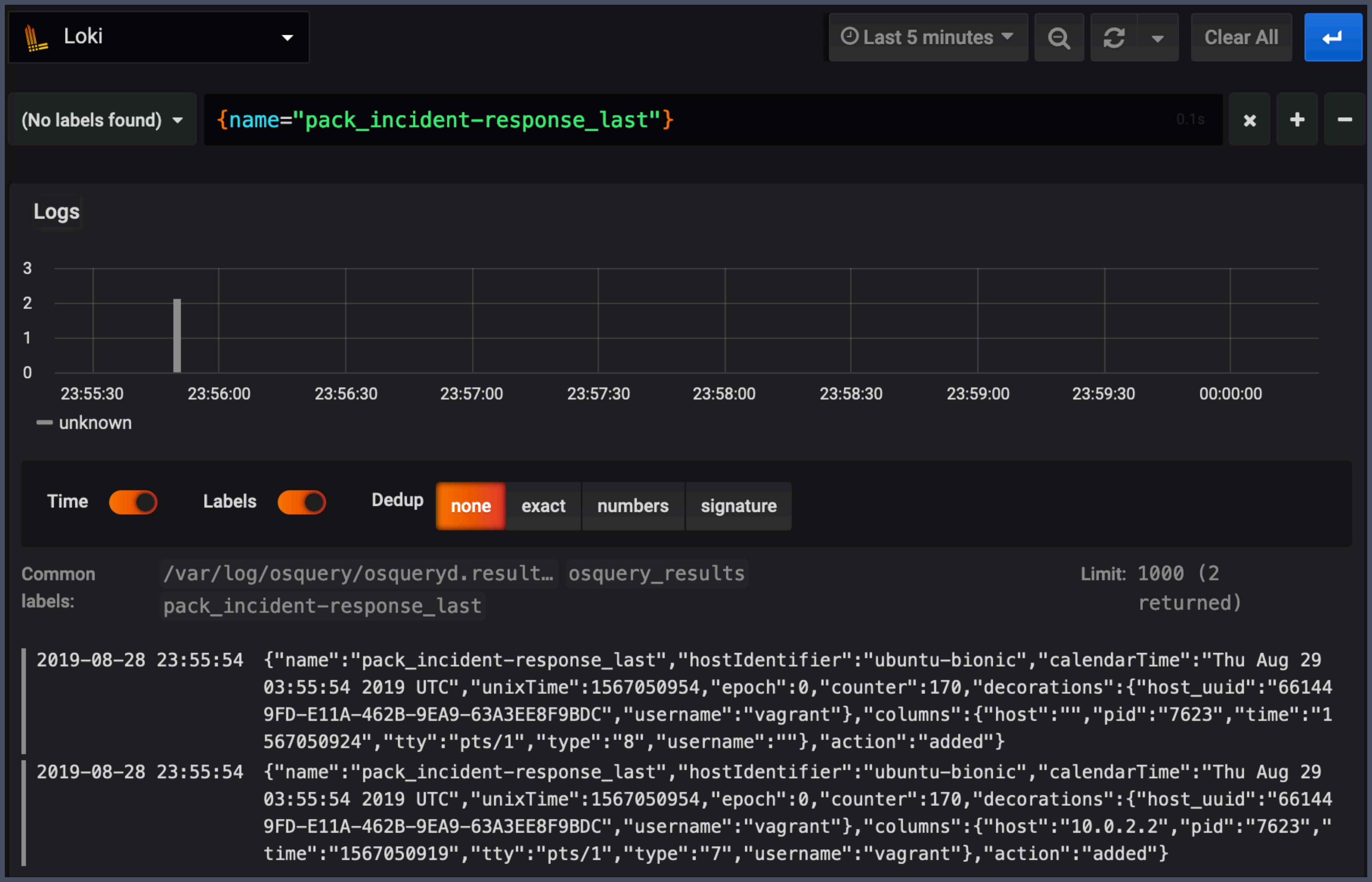
WHAT'S LEFT

CHARTING & ALERTING

GRAFANA

SUPPORTS BOTH PROMTHEUS AND LOKI AS DATA SOURCES

GRAFANA - LOKI



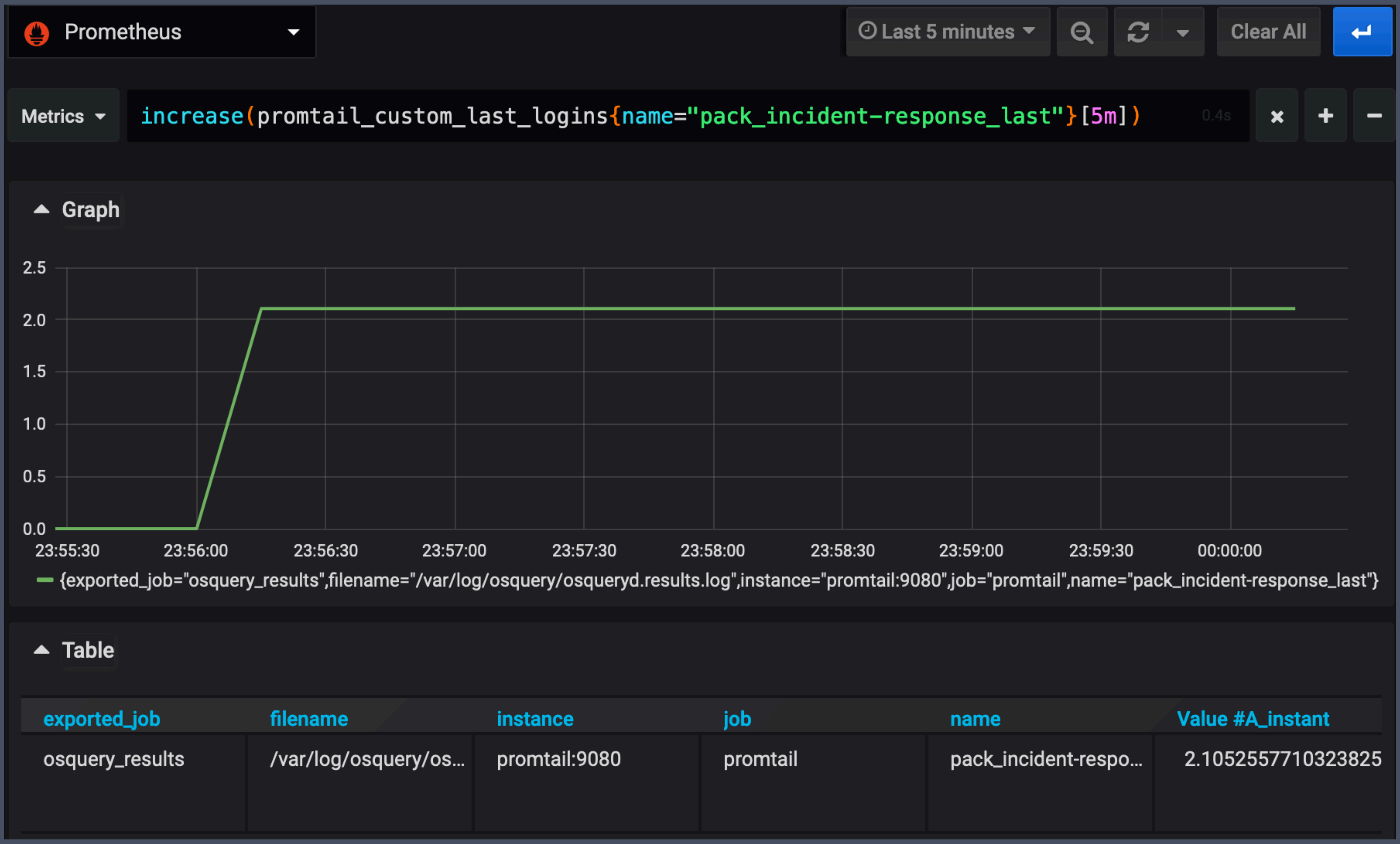
```
{name="pack_incident-response_last"}
```

Common labels: `/var/log/osquery/osqueryd.result…` `osquery_results` `pack_incident-response_last`

Limit: 1000 (2 returned)

```
2019-08-28 23:55:54 {"name":"pack_incident-response_last","hostIdentifier":"ubuntu-bionic","calendarTime":"Thu Aug 29
                    03:55:54 2019 UTC","unixTime":1567050954,"epoch":0,"counter":170,"decorations":{"host_uuid":"66144
                    9FD-E11A-462B-9EA9-63A3EE8F9BDC","username":"vagrant"},"columns":{"host":"","pid":"7623","time":"1
                    567050924","tty":"pts/1","type":"8","username":""},"action":"added"}
2019-08-28 23:55:54 {"name":"pack_incident-response_last","hostIdentifier":"ubuntu-bionic","calendarTime":"Thu Aug 29
                    03:55:54 2019 UTC","unixTime":1567050954,"epoch":0,"counter":170,"decorations":{"host_uuid":"66144
                    9FD-E11A-462B-9EA9-63A3EE8F9BDC","username":"vagrant"},"columns":{"host":"10.0.2.2","pid":"7623","
                    time":"1567050919","tty":"pts/1","type":"7","username":"vagrant"},"action":"added"}
```

GRAFANA - PROMETHEUS

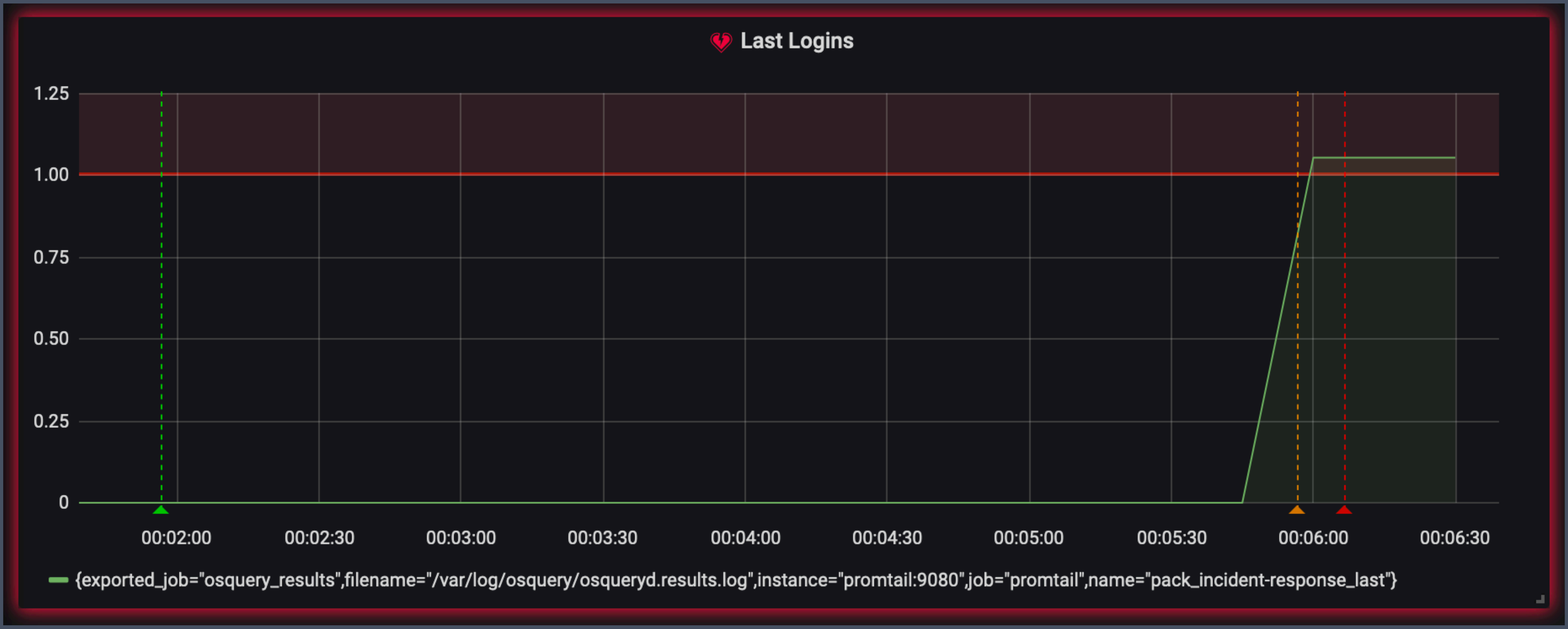


```
increase(promtail_custom_last_logins{name="pack_incident-response_last"}[5m])
```

```
{exported_job="osquery_results",filename="/var/log/osquery/osqueryd.results.log",instance="promtail:9080",job="promtail",name="pack_incident-response_last"}
```

exported_job	filename	instance	job	name	Value #A_instant
osquery_results	/var/log/osquery/os...	promtail:9080	promtail	pack_incident-respo...	2.1052557710323825

GRAFANA - ALERTING



{exported_job="osquery_results",filename="/var/log/osquery/osqueryd.results.log",instance="promtail:9080",job="promtail",name="pack_incident-response_last"}

PUTTING IT ALL TOGETHER

TODO: diagram all the components