$> t3chfest

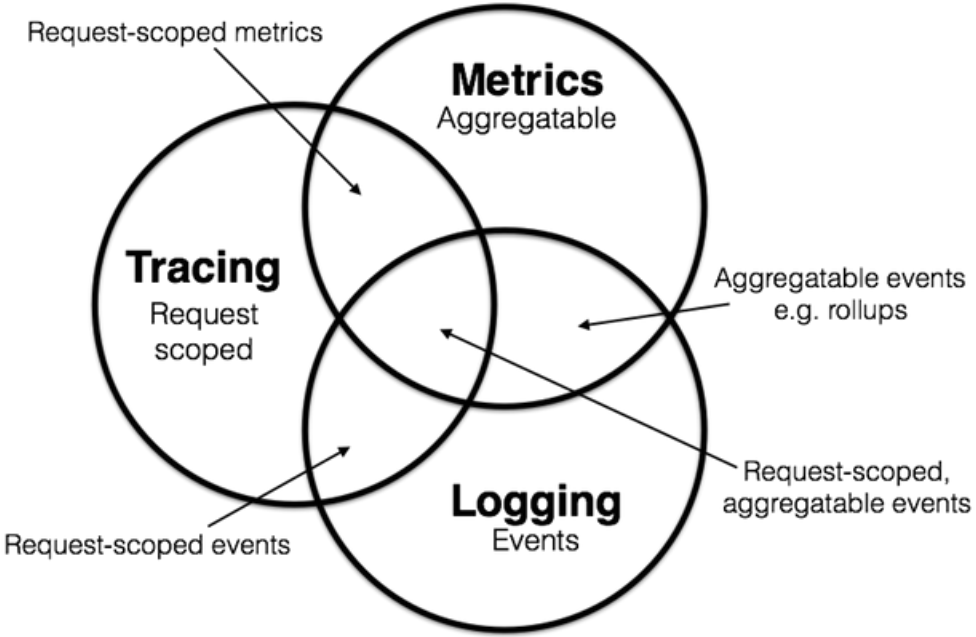
4 logging and metrics systems in 40 minutes

https://speakerdeck.com/lekum/t3chfest-4-logging-and-metrics-systems-in-40-minutes

CARTO

Alejandro Guirao
@lekum
lekum.org
github.com/lekum

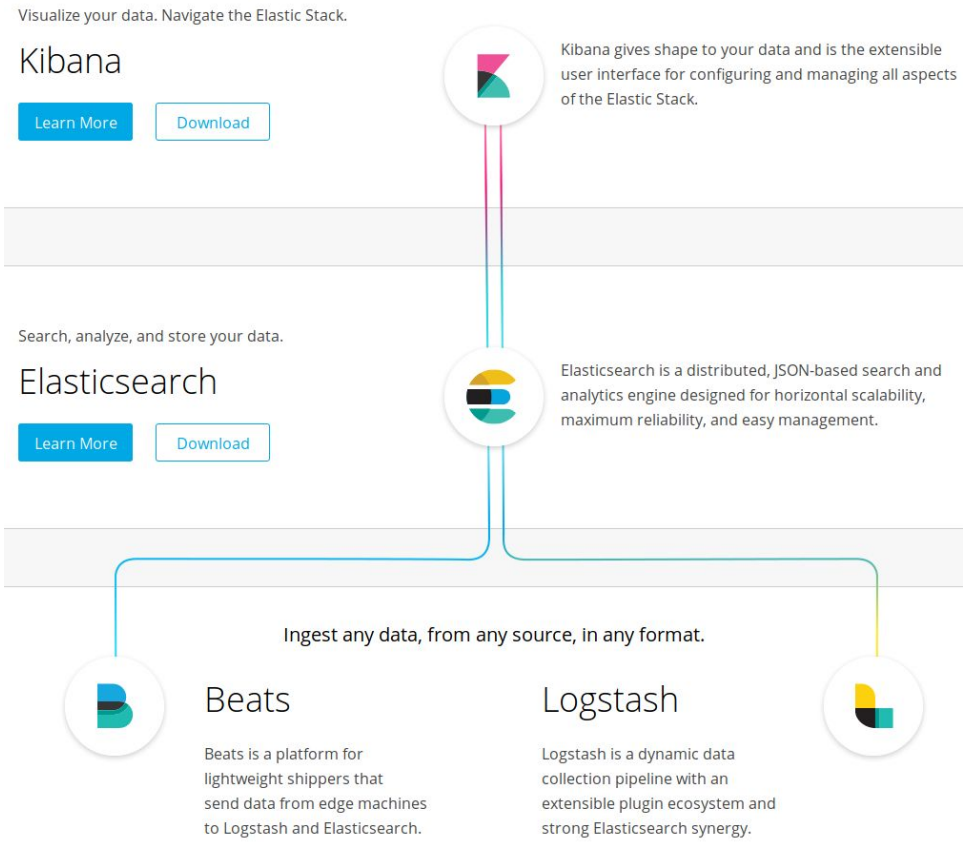
Observability



https://peter.bourgon.org/blog/2017/02/21/metrics-tracing-and-logging.html

Elasticsearch & Friends

Elastic ecosystem



elastic

Elasticsearch


```
curl -H "Content-Type: application/json" -XGET
'http://localhost:9200/social-*/_search' -d '{
 "query": {
   "match": {
     "message": "myProduct"
   }
 },
 "aggregations": {
   "top_10_states": {
     "terms": {
       "field": "state",
       "size": 10
     }
   }
 }
}'
```

elastic

Elasticsearch


```
{
  "hits":{
        "total" : 329,
        "hits" : [
            {
                "_index" : "social-2018",
                "_type" : "_doc",
                "_id" : "0",
                "_score": 1.3862944,
                "_source" : {
                    "user" : "kimchy",
                    "state" : "ID",
                    "date" : "2018-10-15T14:12:12",
                    "message" : "try my product”,
                    "likes": 0
          [...]
```

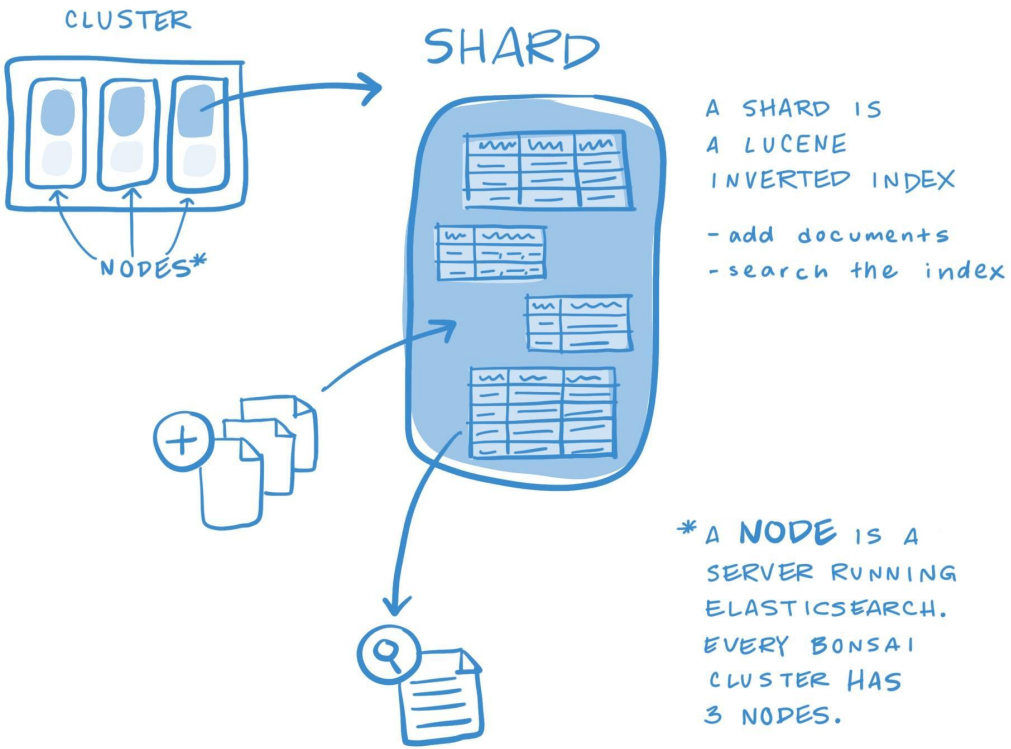
elastic

Elasticsearch


```
{
[...]
  "aggregations" : {
    "top_10_states" : {
      "buckets" : [ {
        "key" : "ID",
        "doc_count" : 27
      },[...]
      }, {
        "key" : "MO",
        "doc_count" : 20
      } ]
    }
  }
}
```

elastic

Elasticsearch architecture



https://docs.bonsai.io/docs/what-are-shards-and-replicas

elastic

Kibana

$> t3chfest

Apache - Total Visitors

4,931,584

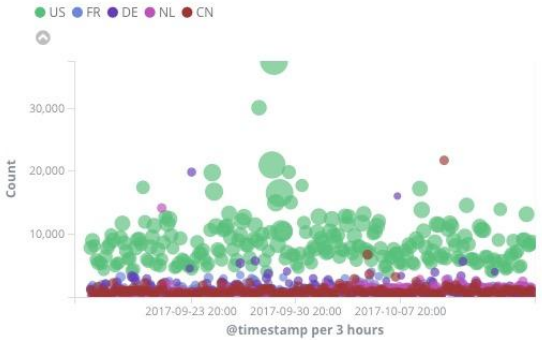
Apache - Unique Visitors

29,740

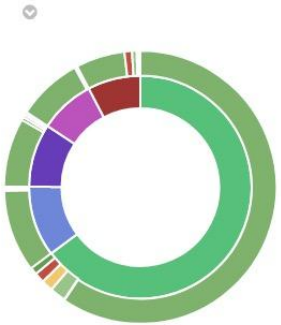
Apache - Unique Visitors ...

City	Unique Visitors
Beijing	562
Redmond	445
Ashburn	400
Chicago	373
Los Angeles	245
Seattle	233
San Jose	232
Singapore	208

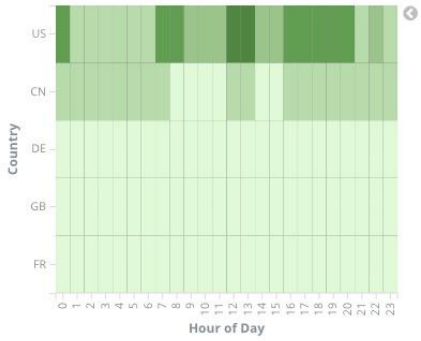
Apache - Bytes and Count



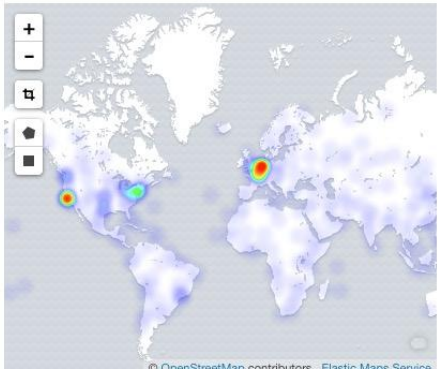
Apache - Country and Status



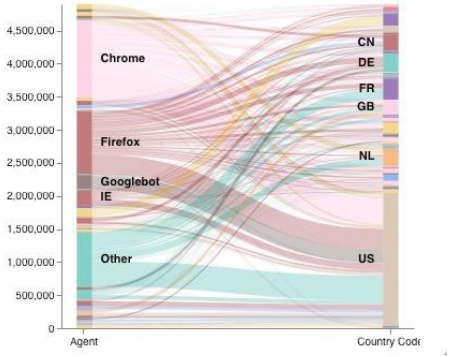
Apache - Country traffic by hour



Apache - Visitor Map (geocentroid)



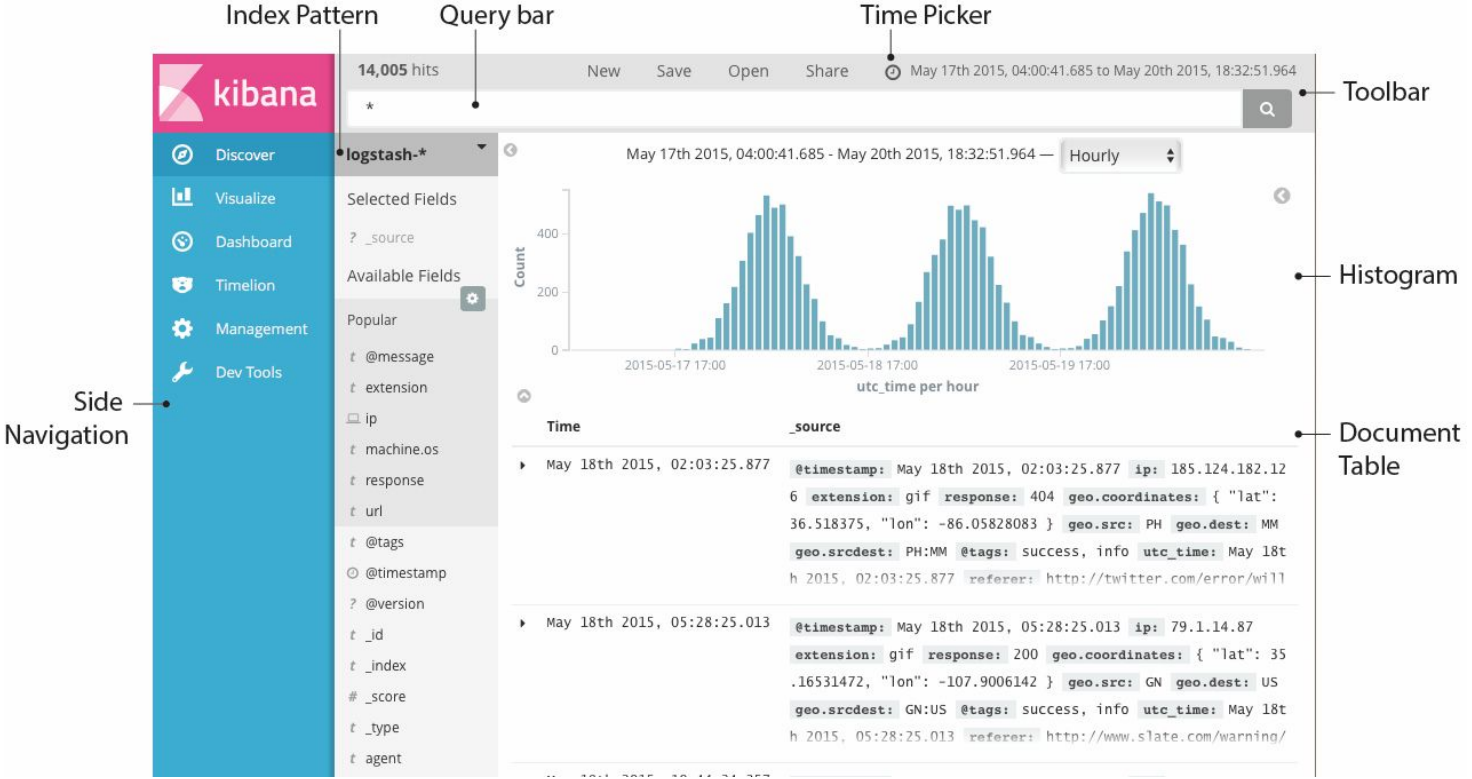
Apache - Browser to Country (vega)



elastic

Kibana - Discover

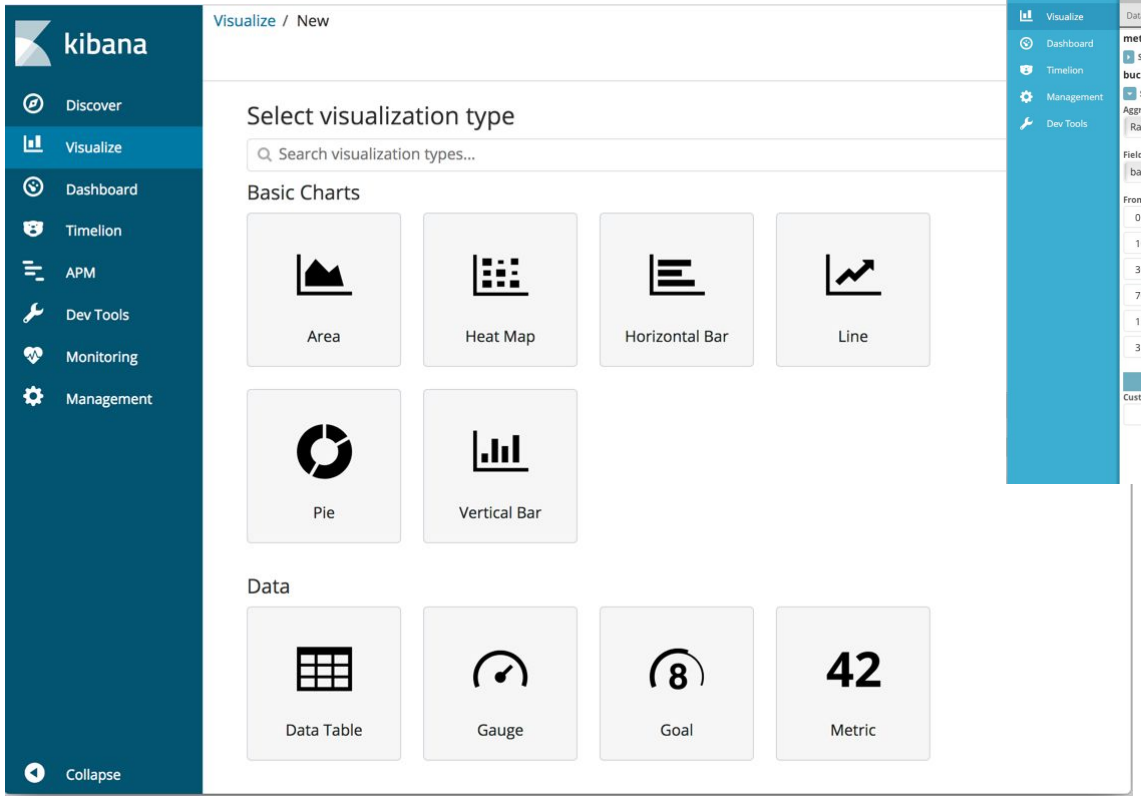
$> t3chfest

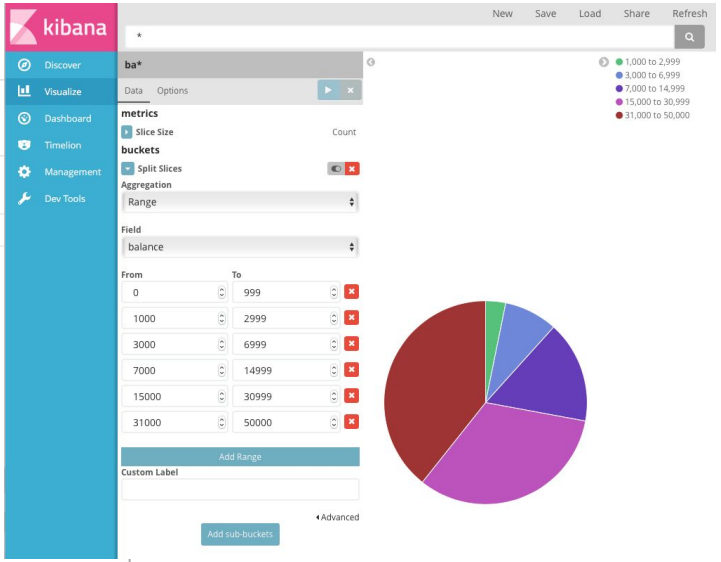


elastic

Kibana - Visualize

$> t3chfest

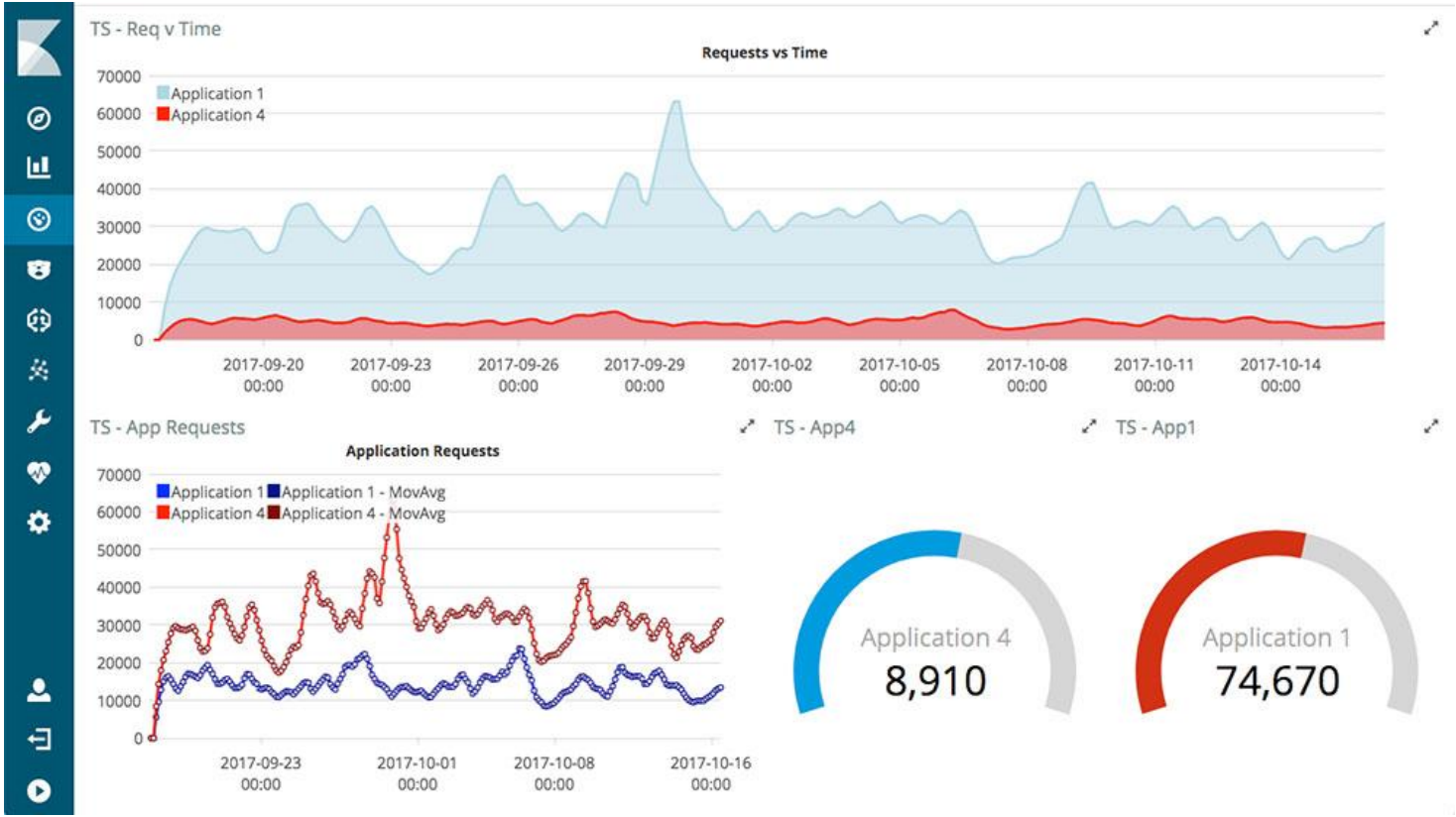




elastic

Kibana - Dashboard

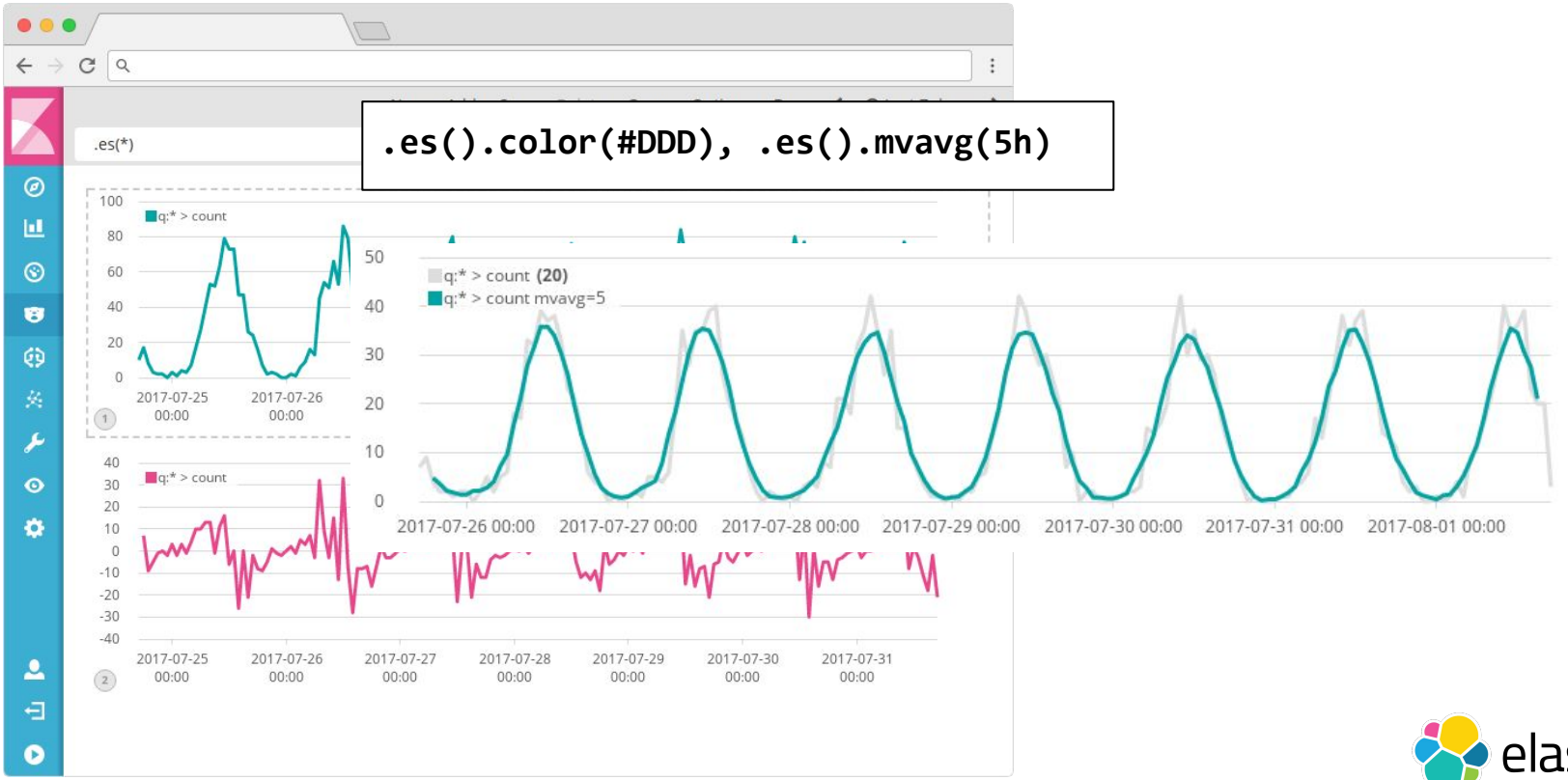
$> t3chfest



elastic

Kibana - Timelion

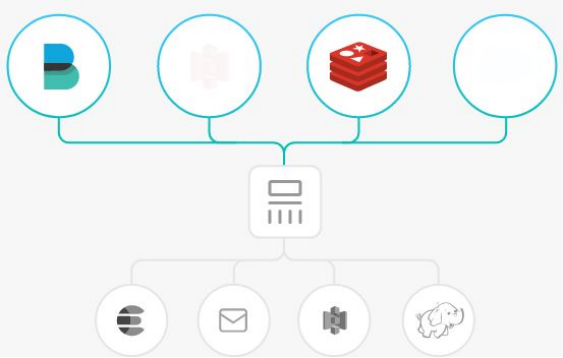
$> t3chfest



elastic

Logstash - Inputs

$> t3chfest



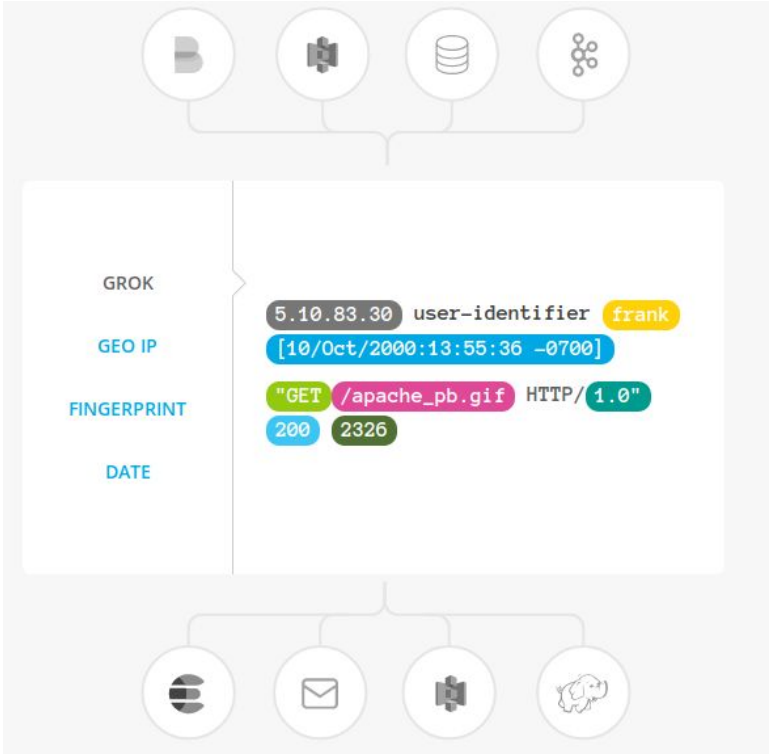
```
redis {
   port => "6379"
   host => "redis.example.com"
   key => "logstash"
   data_type => "list"
 }
```

azure_event_hubs	http	rss	[...]
beats	http_poller	s3	
cloudwatch	imap	salesforce	
couchdb_changes	irc	snmptrap	
dead_letter_queue	jdbc	sqlite	
elasticsearch	jms	sqs	
exec	jmx	stdin	
file	kafka	stomp	
ganglia	kinesis	syslog	
gelf	log4j	tcp	
generator	lumberjack	twitter	
github	meetup	udp	
google_pubsub	pipe	unix	
graphite	puppet_facter	varnishlog	
heartbeat	rabbitmq	websocket	
	redis	xmpp	

elastic

Logstash - Filters

$> t3chfest

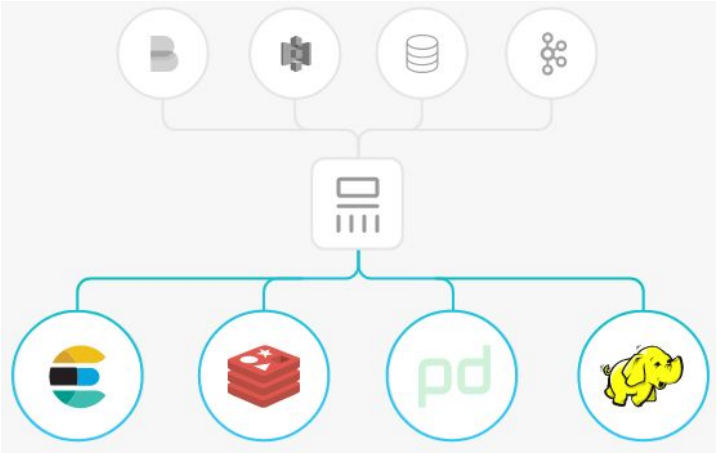



```
filter {
  grok {
    match => { "message" => "%{COMBINEDAPACHELOG}" }
  }
  date {
    match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]
  }
}
```

elastic

Logstash - Outputs

$> t3chfest

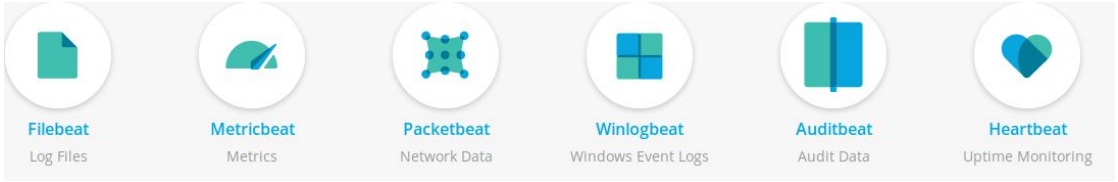


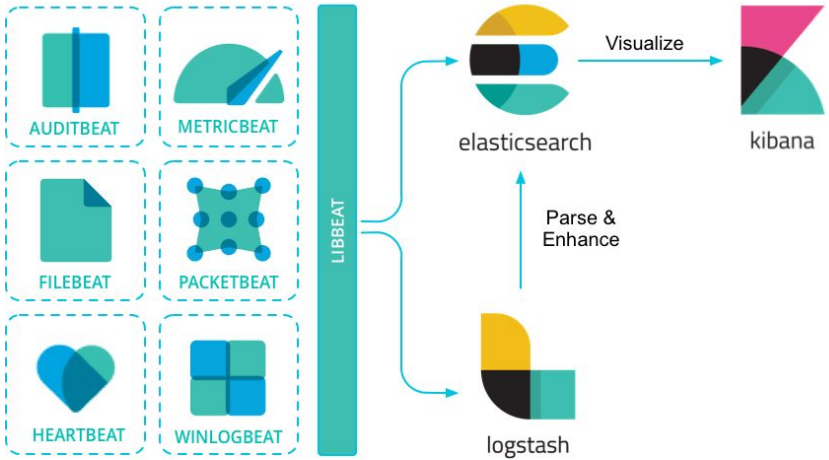
```
output {
  elasticsearch { hosts => ["localhost:9200"] }
  stdout { codec => rubydebug }
}
```

boundary
circonus
cloudwatch
csv
datadog
datadog_metrics
elasticsearch
email
exec
file
ganglia
gelf
google_bigquery
google_pubsub
graphite
kafka
librato
loggly
lumberjack
metriccatcher
mongodb
nagios
nagios_nsca
opentsdb
pagerduty
pipe
rabbitmq
redis
[...]

elastic

Beats






```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/*.log
output.elasticsearch:
  hosts: ["myEShost:9200"]
  username: "filebeat_internal"
  password: "YOUR_PASSWORD"
```

elastic

Deploying Elastic Stack

$> t3chfest

```
version: '2.2'
services:
  elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch:6.6.1
    container_name: elasticsearch
    environment:
      - cluster.name=docker-cluster
      - bootstrap.memory_lock=true
      - "ES_JAVA_OPTS=-Xms512m -Xmx512m"
    volumes:
      - esdata1:/usr/share/elasticsearch/data
    ports:
      - 9200:9200
    networks:
      - esnet
  elasticsearch2:
    image: docker.elastic.co/elasticsearch/elasticsearch:6.6.1
    container_name: elasticsearch2
   [...]
```

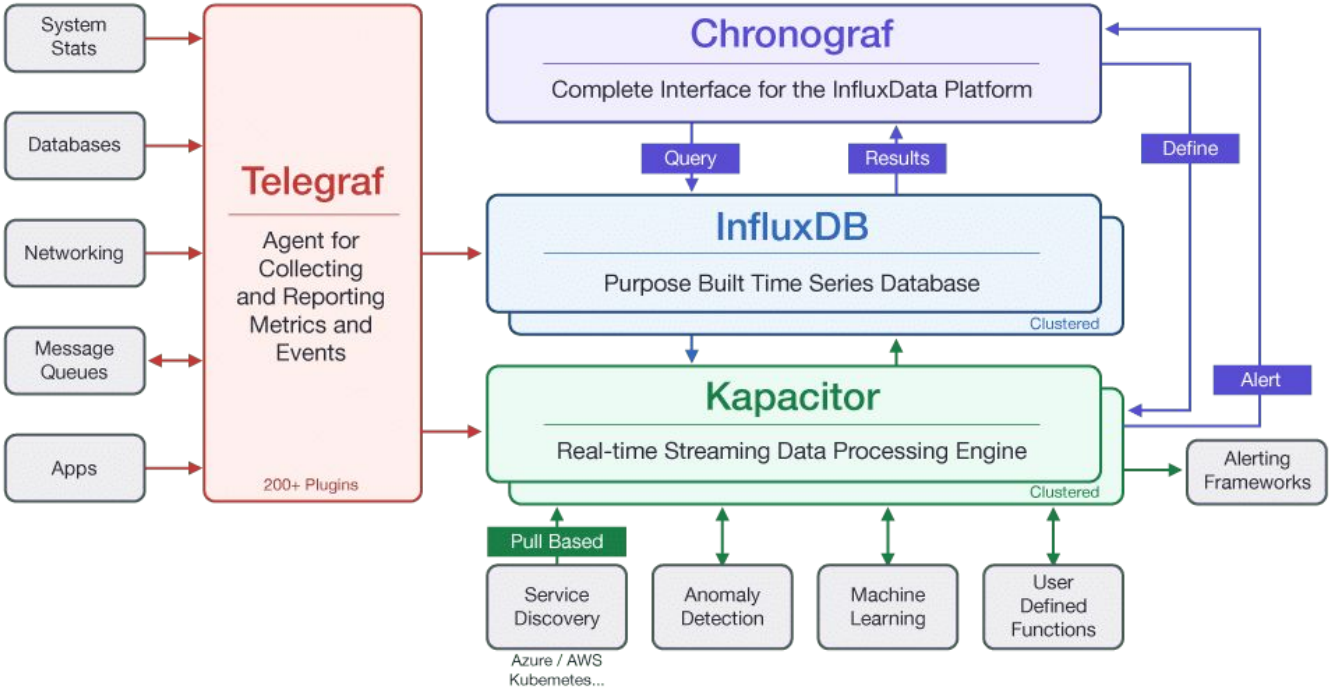
- zip/tar.gz
- deb
- rpm
- msi
- docker

elastic

InfluxDB & Friends

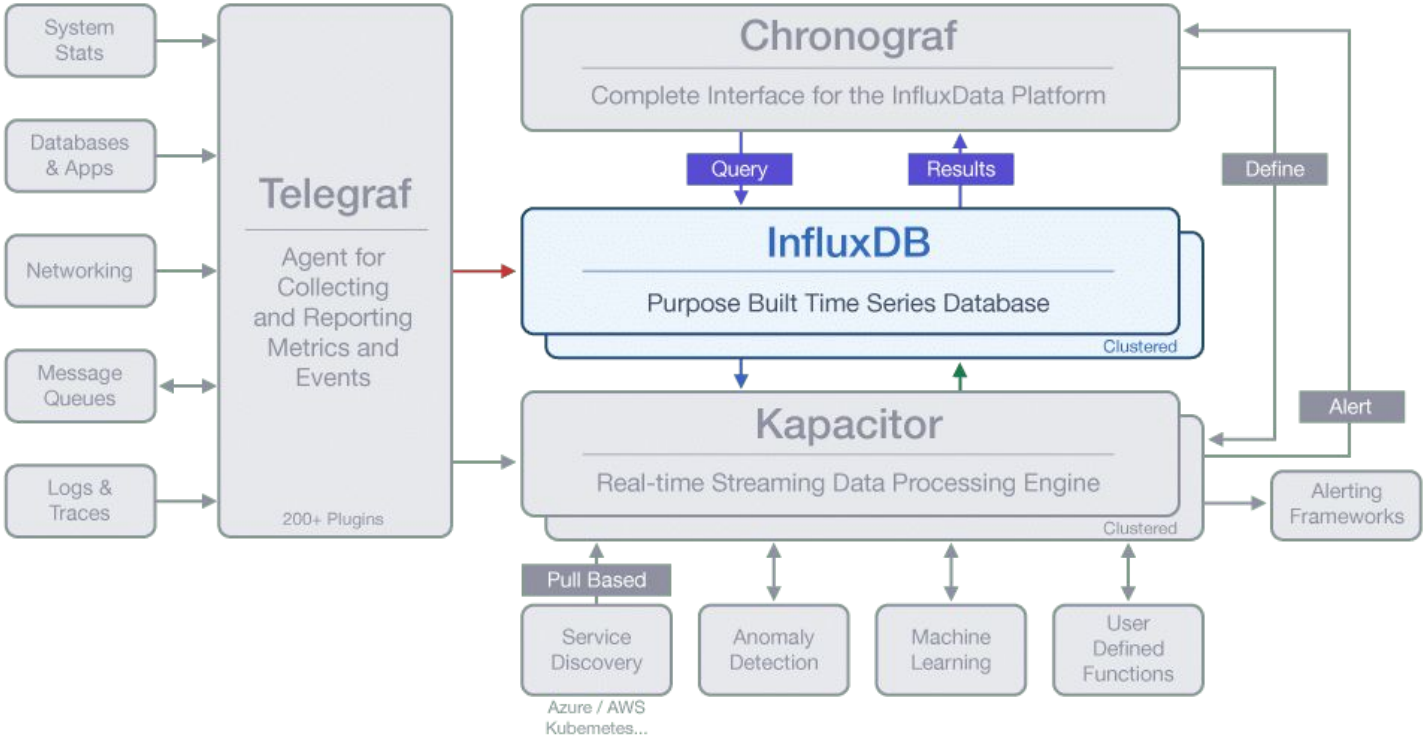
TICK stack

$> t3chfest



influxdata

InfluxDB



influxdata

InfluxDB - Time Series Database

$> t3chfest

```
<measurement>[,<tag-key>=<tag-value>...] <field-key>=<field-value>[,<field2-key>=<field2-value>...]
[unix-nano-timestamp]
```

```
cpu,host=serverA,region=us_west value=0.64
payment,device=mobile,product=Notepad,method=credit billed=33,licenses=3i
1434067467100293230
stock,symbol=AAPL bid=127.46,ask=127.48
temperature,machine=unit42,type=assembly external=25,internal=37 1434067467000000000
```

influxdata

InfluxDB - Time Series Database

$> t3chfest

```
<measurement>[,<tag-key>=<tag-value>...] <field-key>=<field-value>[,<field2-key>=<field2-value>...]
[unix-nano-timestamp]
```

```
cpu,host=serverA,region=us_west value=0.64
payment,device=mobile,product=Notepad,method=credit billed=33,licenses=3i
1434067467100293230
stock,symbol=AAPL bid=127.46,ask=127.48
temperature,machine=unit42,type=assembly external=25,internal=37 1434067467000000000
```

influxdata

InfluxDB - Time Series Database

$> t3chfest

```
<measurement>[,<tag-key>=<tag-value>...] <field-key>=<field-value>[,<field2-key>=<field2-value>...]
[unix-nano-timestamp]
```

```
cpu,host=serverA,region=us_west value=0.64
payment,device=mobile,product=Notepad,method=credit billed=33,licenses=3i
1434067467100293230
stock,symbol=AAPL bid=127.46,ask=127.48
temperature,machine=unit42,type=assembly external=25,internal=37 1434067467000000000
```

influxdata

InfluxDB - Time Series Database

$> t3chfest

```
<measurement>[,<tag-key>=<tag-value>...] <field-key>=<field-value>[,<field2-key>=<field2-value>...]
[unix-nano-timestamp]
```

```
cpu,host=serverA,region=us_west value=0.64
payment,device=mobile,product=Notepad,method=credit billed=33,licenses=3i
1434067467100293230
stock,symbol=AAPL bid=127.46,ask=127.48
temperature,machine=unit42,type=assembly external=25,internal=37 1434067467000000000
```

influxdata

InfluxDB - Time Series Database

$> t3chfest

```
<measurement>[,<tag-key>=<tag-value>...] <field-key>=<field-value>[,<field2-key>=<field2-value>...]
[unix-nano-timestamp]
```

```
cpu,host=serverA,region=us_west value=0.64
payment,device=mobile,product=Notepad,method=credit billed=33,licenses=3i
1434067467100293230
stock,symbol=AAPL bid=127.46,ask=127.48
temperature,machine=unit42,type=assembly external=25,internal=37 1434067467000000000
```

influxdata

InfluxDB - Time Series Database


```
$ influx -precision rfc3339
> CREATE DATABASE mydb
> SHOW DATABASES
name: databases
---------------
name
_internal
mydb

> USE mydb
Using database mydb
```

InfluxDB - Time Series Database

$> t3chfest

```
> INSERT cpu,host=serverA,region=us_west value=0.64
>
> SELECT "host", "region", "value" FROM "cpu"
name: cpu
---------
time                                host     region   value
2015-10-21T19:28:07.580664347Z  serverA     us_west    0.64
>
> INSERT temperature,machine=unit42,type=assembly external=25,internal=37
>
> SELECT * FROM "temperature"
name: temperature
-----------------
time                                external     internal   machine    type
2015-10-21T19:28:08.385013942Z  25              37               unit42  assembly
```

influxdata

InfluxDB - InfluxQL functions

$> t3chfest

```
COUNT()
DISTINCT()
INTEGRAL()
MEAN()
MEDIAN()
MODE()
SPREAD()
STDDEV()
SUM()
```

```
BOTTOM()
FIRST()
LAST()
MAX()
MIN()
PERCENTILE()
SAMPLE()
TOP()
```

```
CHANDE_MOMENTUM_OSCILLATOR()
EXPONENTIAL_MOVING_AVERAGE()
DOUBLE_EXPONENTIAL_MOVING_AVERAGE()
KAUFMANS_EFFICIENCY_RATIO()
KAUFMANS_ADAPTIVE_MOVING_AVERAGE()
TRIPLE_EXPONENTIAL_MOVING_AVERAGE()
TRIPLE_EXPONENTIAL_DERIVATIVE()
RELATIVE_STRENGTH_INDEX()
```

```
ABS()
ACOS()
ASIN()
ATAN()
ATAN2()
CEIL()
COS()
CUMULATIVE_SUM()
DERIVATIVE()
DIFFERENCE()
ELAPSED()
EXP()
FLOOR()
HISTOGRAM()
LN()
LOG()
```

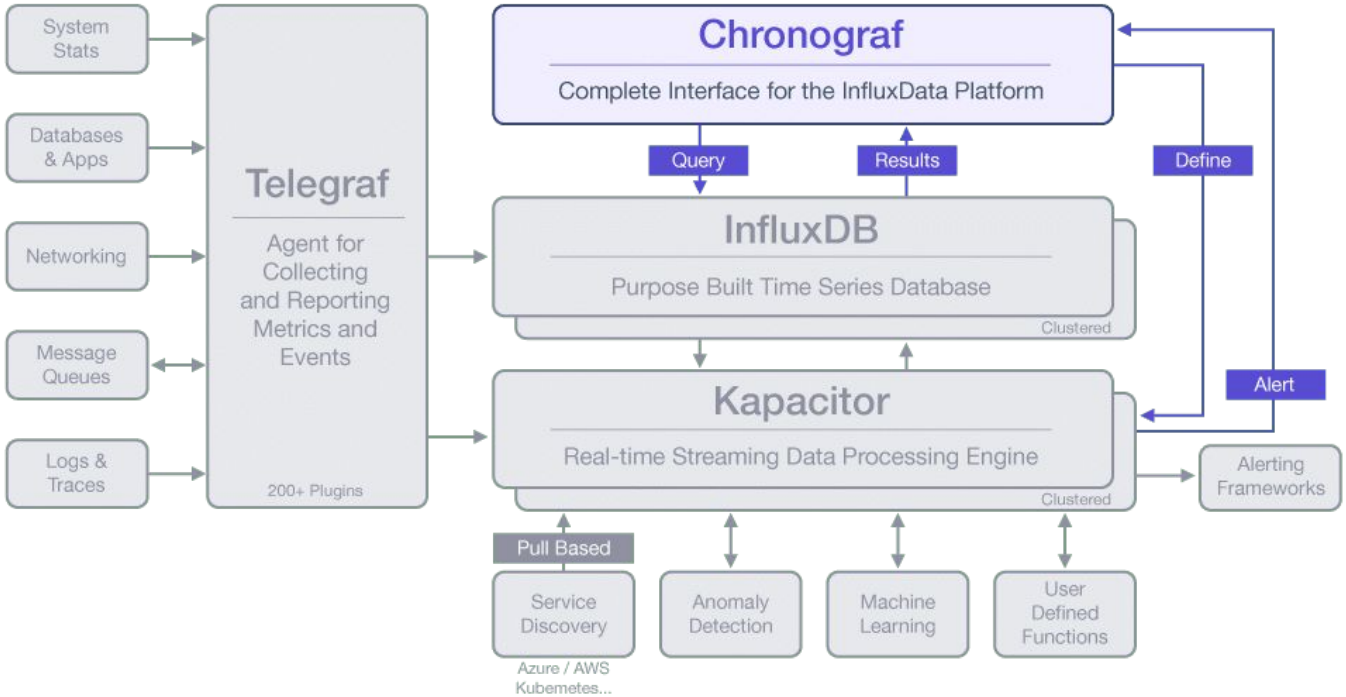
```
LOG2()
LOG10()
MOVING_AVERAGE()
NON_NEGATIVE_DERIVATIVE()
NON_NEGATIVE_DIFFERENCE()
POW()
ROUND()
SIN()
SQRT()
TAN()
```

influxdata

InfluxDB - Features

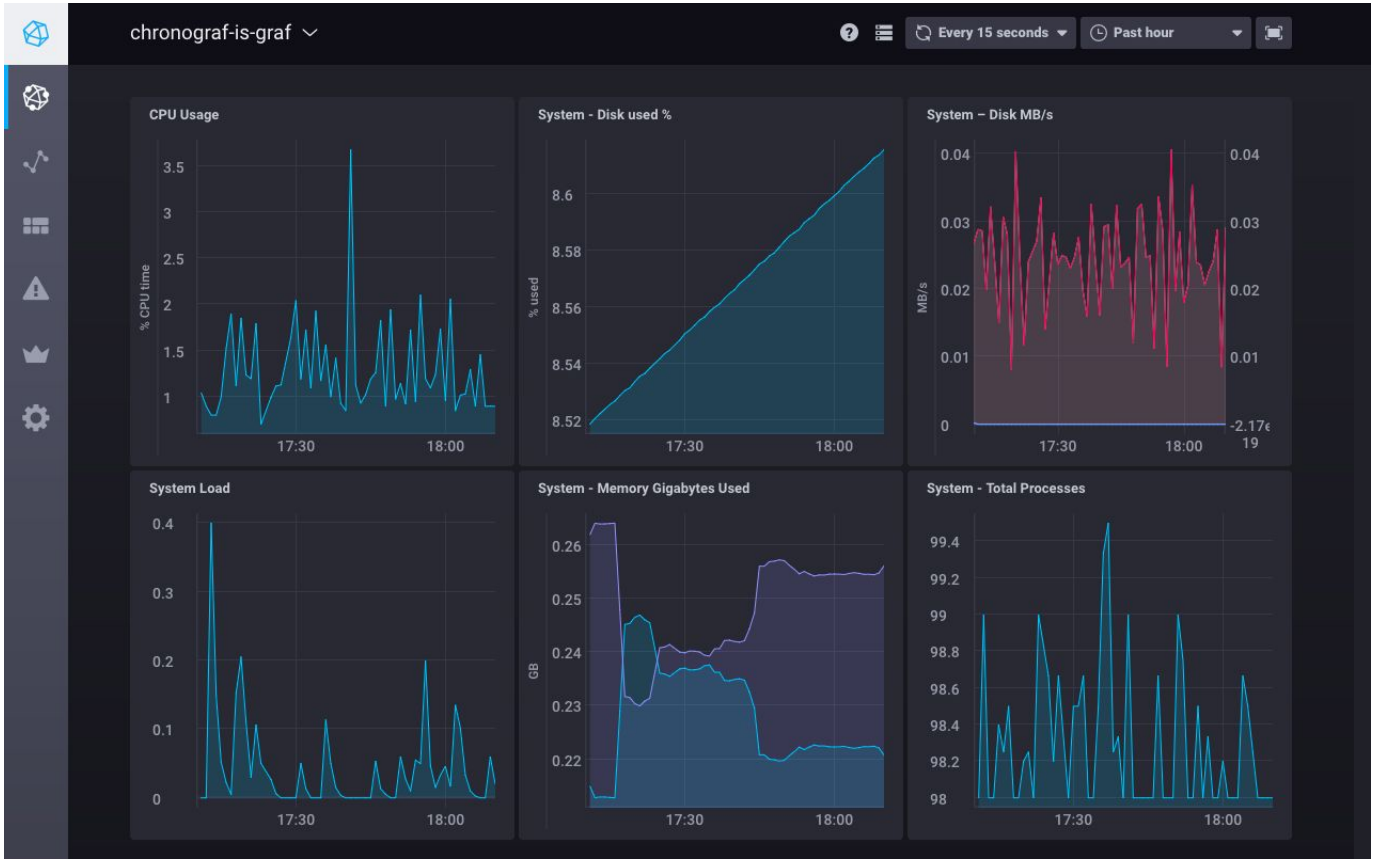
- Retention policy (DURATION and REPLICATION)
- Continuous Queries
- Not a full CRUD database but more like a CR-ud

Cronograf



influxdata

Chronograf



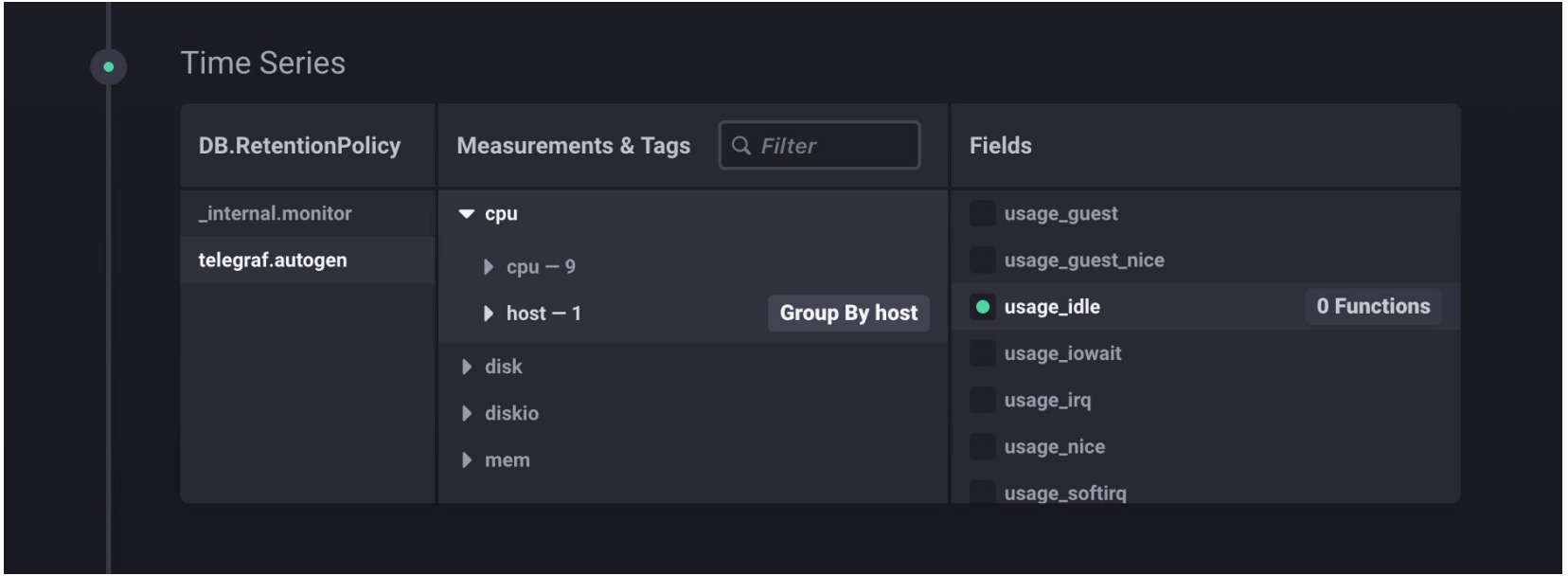
influxdata

Chronograf

$> t3chfest

influxdata

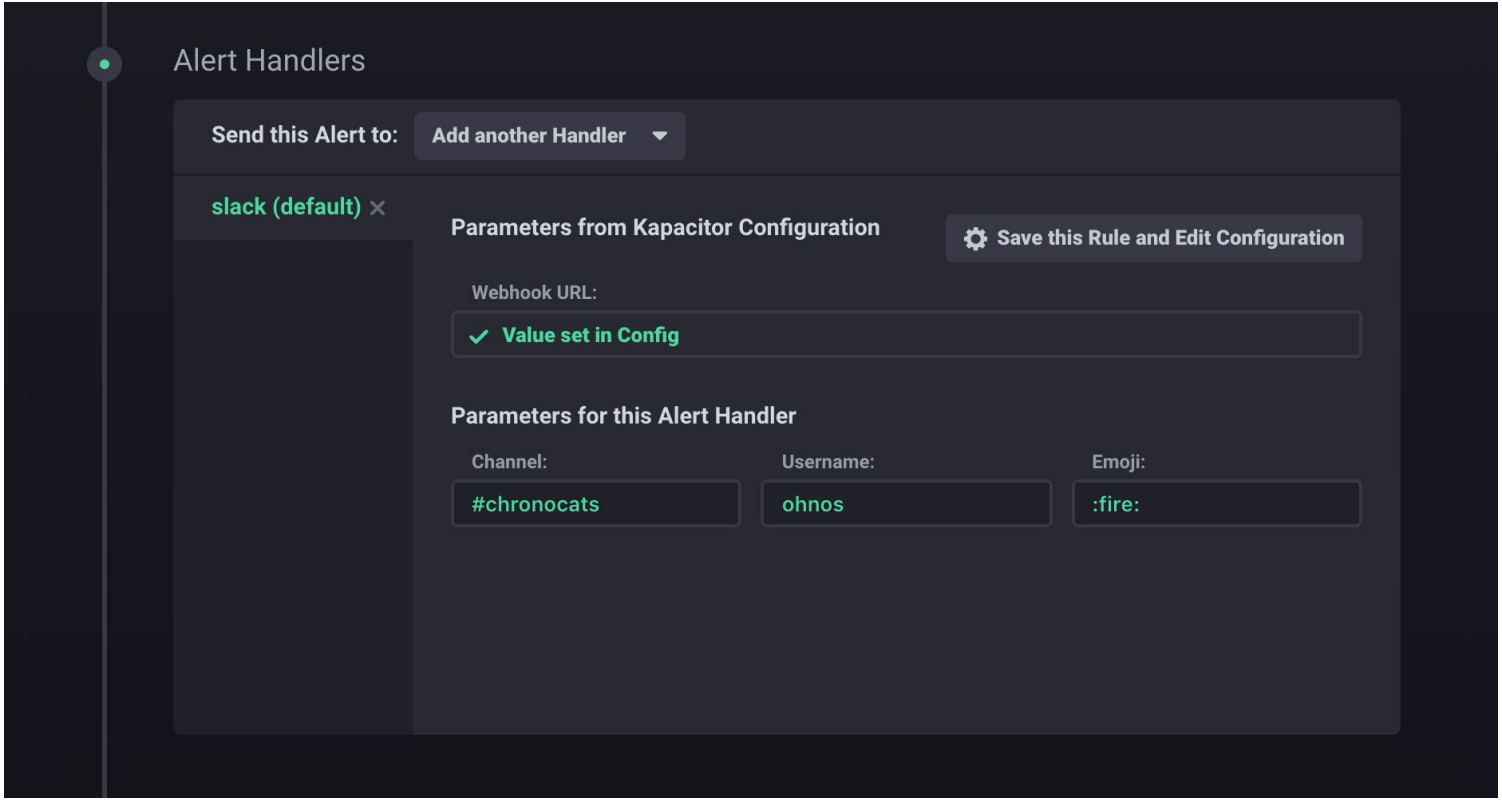
Chronograf


Time Series
DB.RetentionPolicy
Measurements & Tags
Filter
Fields
_internal.monitor
telegraf.autogen
cpu
cpu — 9
host — 1
Group By host
disk
diskio
mem
usage_guest
usage_guest_nice
usage_idle
0 Functions
usage_iowait
usage_irq
usage_nice
usage_softirq

Chronograf


Conditions
Send Alert where
usage_idle
is
less than
80
Preview Data from
Past 15m
100
90
80
70
cpu.usage_idle
16:42
16:44
16:46
16:48
16:50
16:52
16:54
16:56

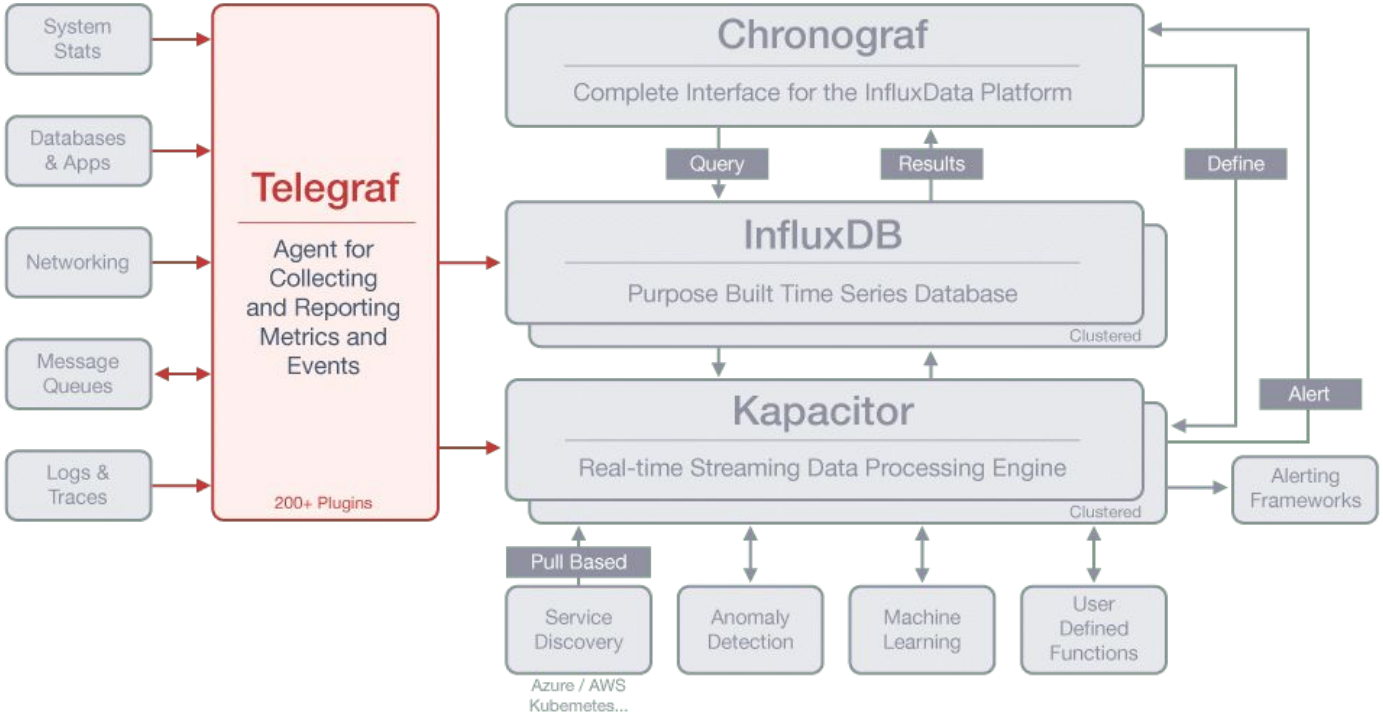
Chronograf


Alert Handlers
Send this Alert to:
Add another Handler
slack (default)
Parameters from Kapacitor Configuration
Save this Rule and Edit Configuration
Webhook URL:
Value set in Config
Parameters for this Alert Handler
Channel:
#chronocats
Username:
ohnos
Emoji:
:fire:

Chronograf


Message
Your idle CPU usage is {{.Level}} at {{ index .Fields "value" }}.
Templates:
{{.ID}}
{{.Name}}
{{.TaskName}}
{{.Group}}
{{.Tags}}
{{.Level}}
{{ index .Fields "value" }}
{{.Time}}

Telegraf



influxdata

Telegraf - Plugins

- More than 100 input plugins
  - statsd, phpfpm, twemproxy, zipkin, postfix, nginx, tengine, rethinkdb, http, passenger, icinga2, nvidia_smi, kibana, consul, mysql, aerospike, mcrouter, kubernetes, linux_sysctl_fs, kernel, file, udp_listener, cpu, sysstat…
- Outputs plugins
  - amon, amqp, application_insights, azure_monitor, cloudwatch, cratedb, datadog, discard, elasticsearch, file, graphite, graylog, http, influxdb, influxdb_v2, instrumental, kafka, kinesis, librato, mqtt, nats, nsq, opentsdb, prometheus_client, riemann, riemann_legacy, socket_writer, stackdriver, wavefront

Telegraf - Plugins

- Processor plugins
  - converter, enum, override, parser, printer, regex, rename, strings, topk
- Aggregator plugins
  - BasicStats, Histogram, MinMax, ValueCounter

Telegraf - Configuration


```
$ telegraf --input-filter cpu:mem:net:swap --output-filter influxdb:kafka config > telegraf.conf
```

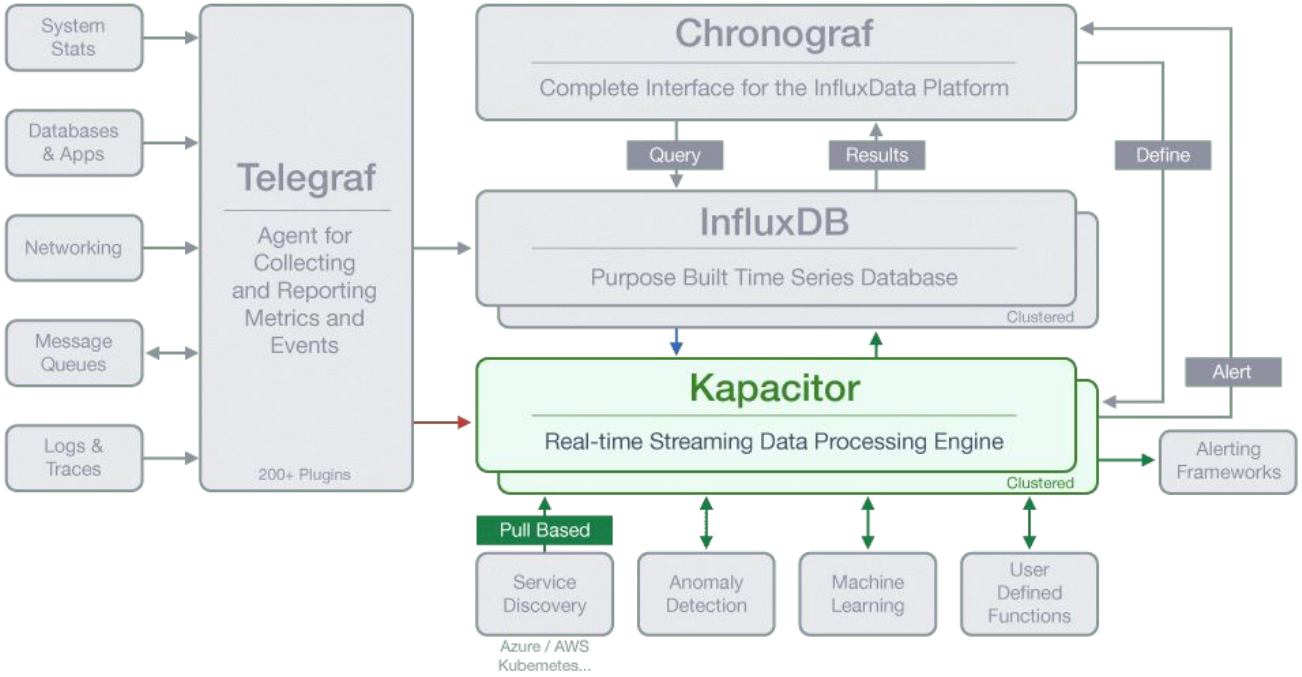
```
[global_tags]
  dc = "denver-1"

[agent]
  interval = "10s"

# OUTPUTS
[[outputs.influxdb]]
  url = "http://192.168.59.103:8086" # required.
  database = "telegraf" # required.

# INPUTS
[[inputs.cpu]]
  percpu = true
  totalcpu = false
  # filter all fields beginning with 'time_'
  fielddrop = ["time_*"]
```

Kapacitor



influxdata

Kapacitor - Stream

cpu_alert.tick

```
dbrp "telegraf"."autogen"

stream
    // Select just the cpu measurement from our example database.
    |from()
        .measurement('cpu')
    |alert()
        .crit(lambda: int("usage_idle") <  70)
        // Whenever we get an alert write it to a file.
        .log('/tmp/alerts.log')
```

```
$ kapacitor define cpu_alert -tick cpu_alert.tick
$ kapacitor enable cpu_alert
```

Kapacitor - Stream

$> t3chfest

```
stream
    |from()
        .measurement('cpu')
    // create a new field called 'used' which inverts the idle cpu.
    |eval(lambda: 100.0 - "usage_idle")
        .as('used')
    |groupBy('service', 'datacenter')
    |window()
        .period(1m)
        .every(1m)
    // calculate the 95th percentile of the used cpu.
    |percentile('used', 95.0)
    |eval(lambda: sigma("percentile"))
        .as('sigma')
        .keep('percentile', 'sigma')
    |alert()
        .id('{{ .Name }}/{{ index .Tags "service" }}/{{ index .Tags "datacenter"}}')
        .message('{{ .ID }} is {{ .Level }} cpu-95th:{{ index .Fields "percentile" }}')
        // Compare values to running mean and standard deviation
        .warn(lambda: "sigma" > 2.5)
        .crit(lambda: "sigma" > 3.0)
        .log('/tmp/alerts.log')

        // Send alerts to slack
        .slack()
        .channel('#alerts')

        // Sends alerts to PagerDuty
        .pagerDuty()
```

influxdata

Kapacitor - Batch


```
dbrp "telegraf"."autogen"

batch
    |query('''
        SELECT mean(usage_idle)
        FROM "telegraf"."autogen"."cpu"
    ''')
        .period(5m)
        .every(5m)
        .groupBy(time(1m), 'cpu')
    |alert()
        .crit(lambda: "mean" < 70)
        .log('/tmp/batch_alerts.log')
```

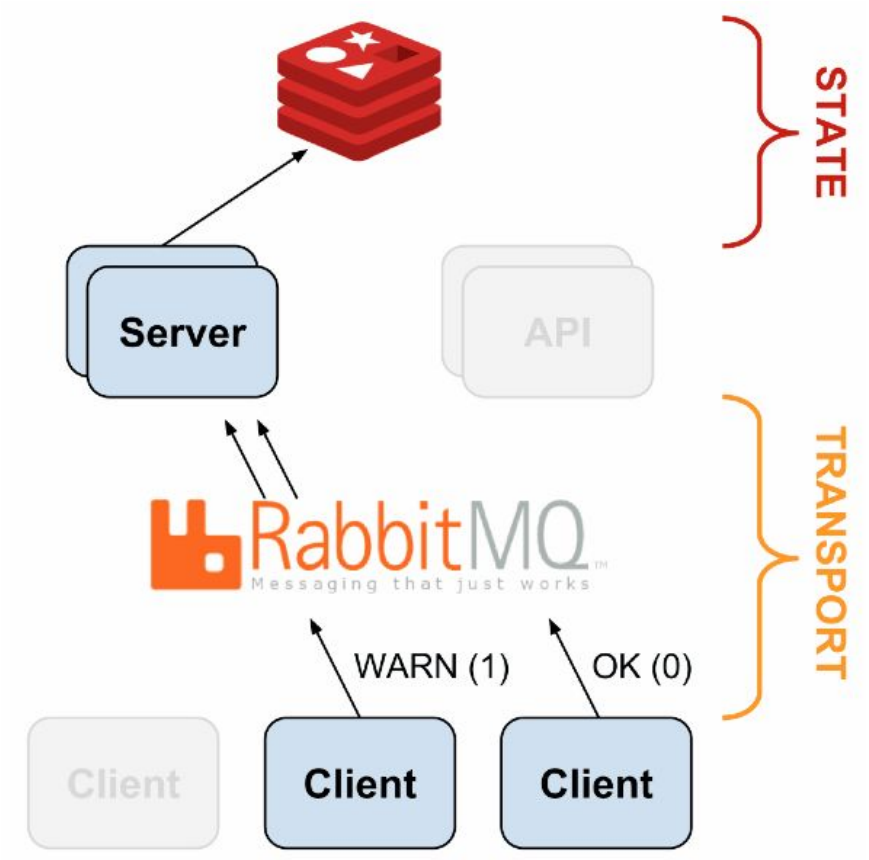
Deploying the TICK stack


```
version: '3'
services:
  influxdb:
      image: "influxdb:latest"
  telegraf:
      image: "telegraf:latest"
      volumes:
      - ./etc/telegraf:/etc/telegraf
  kapacitor:
      image: "kapacitor:latest"
      volumes:
      - ./etc/kapacitor:/etc/kapacitor
      - ./var/log/kapacitor:/var/log/kapacitor
      - ./home/kapacitor:/home/kapacitor
```

- .deb
- .rpm
- MacOS
- Win .exe
- Docker

Sensu

Sensu (legacy) architecture



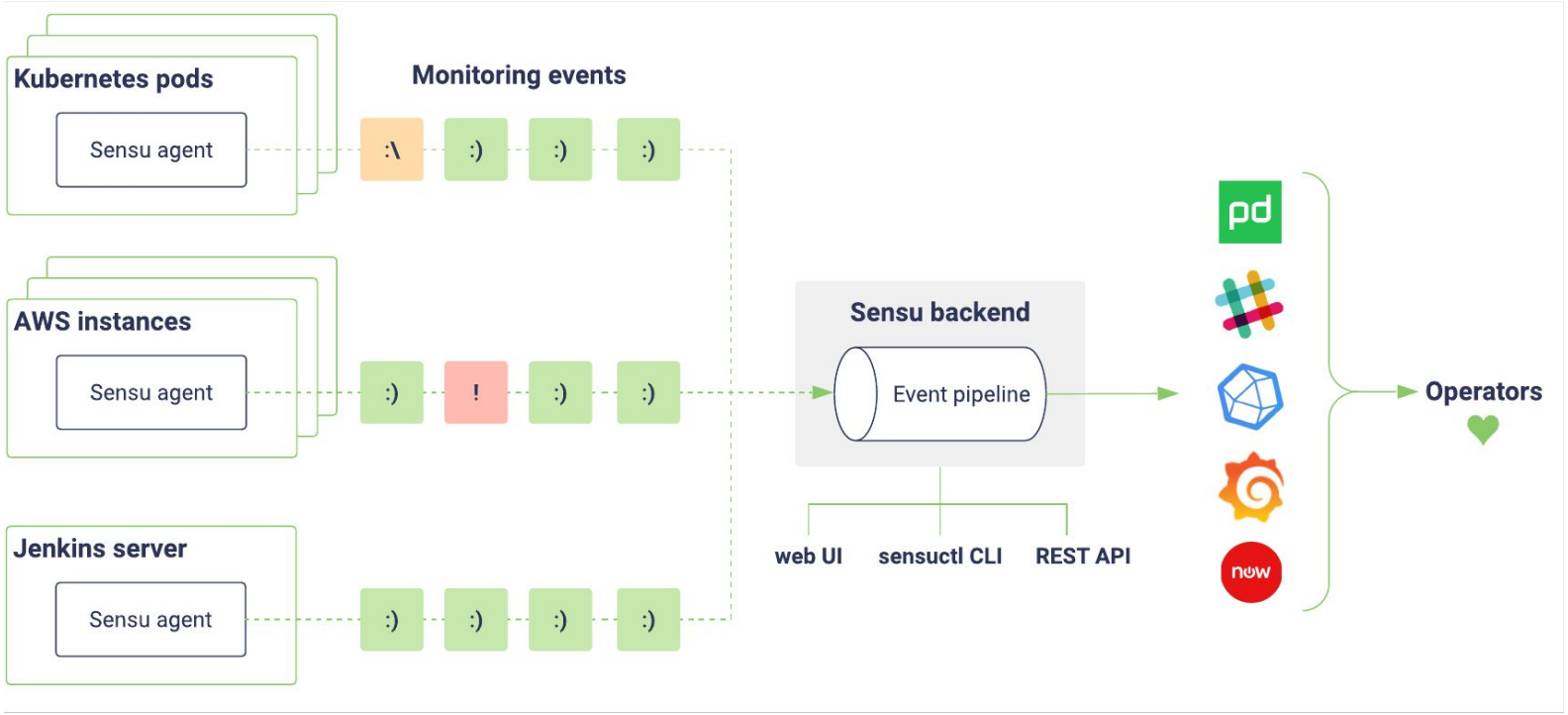
Sensu

Sensu-go

- Implemented in Go
  - sensu-backend
  - sensu-agent
- No need for third-party transport, storage or dashboard
- More powerful API
- CLI
  - sensuctl
- Built-in StatsD metrics collector
- Configuration via YAML
- RBAC

Sensu

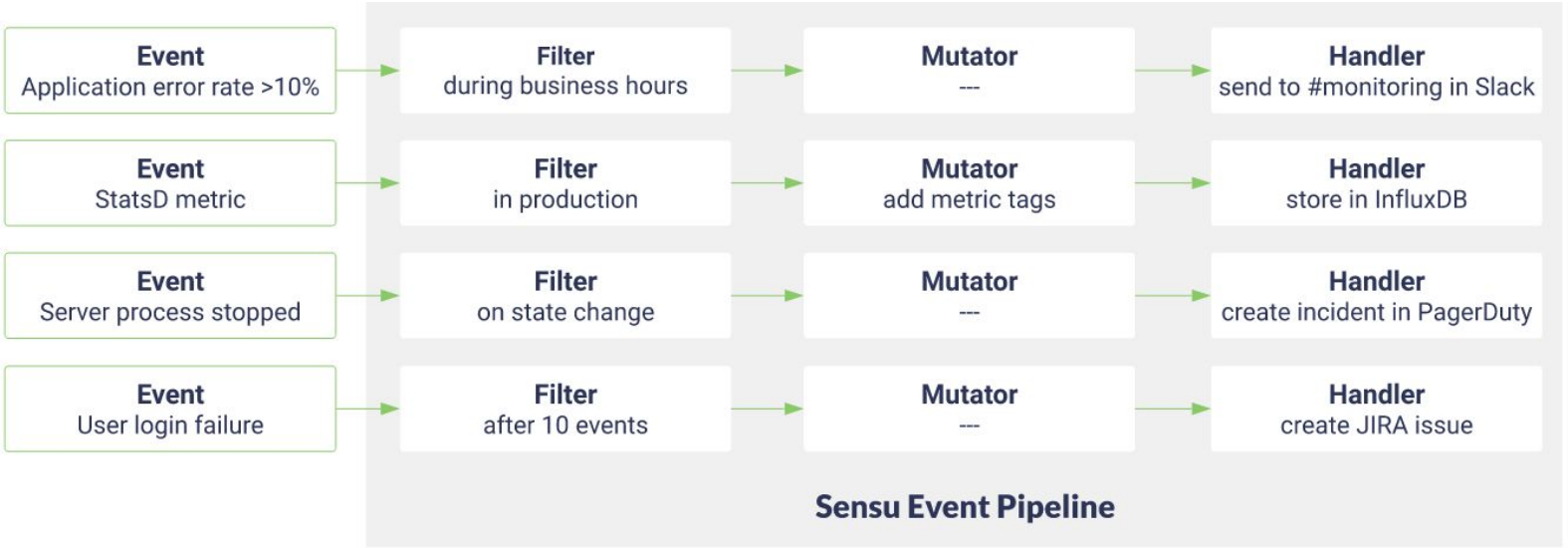
Sensu-go



Sensu

Sensu-go pipeline

$> t3chfest



Sensu

Configuring checks and handlers


```
sensuctl check create check-cpu \
--command 'check-cpu.sh -w 75 -c 90' \
--interval 60 \
--subscriptions linux
```

```
sensuctl handler create influx-db \
--type pipe \
--command "sensu-influxdb-handler \
--addr 'http://123.4.5.6:8086' \
--db-name 'myDB' \
--username 'foo' \
--password 'bar'"
```

Sensu

Hooks and filters


```
sensuctl hook create nginx-restart  \
--command 'sudo systemctl restart nginx' \
--timeout 10
```

```
sensuctl filter create hourly \
  --action allow \
  --statements "event.Check.Occurrences == 1 ||
event.Check.Occurrences % (3600 / event.Check.Interval) == 0"
```

Sensu

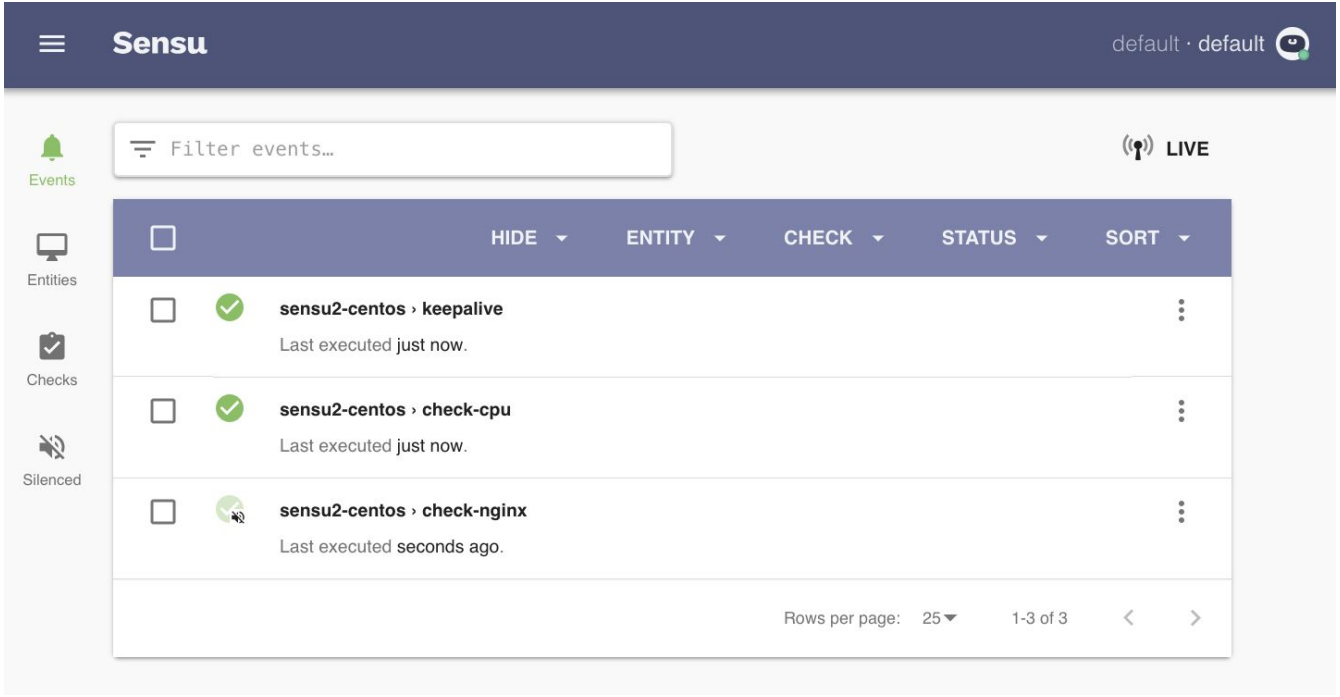
Assets


```
$ sensuctl asset create check_website.tar.gz \
  -u http://example.com/check_website.tar.gz \
  --sha512 "$(sha512sum check_website.tar.gz | cut -f1 -d ' ')"
```

Sensu

Built-in dashboard

$> t3chfest



Sensu

Deploying Sensu-go


```
$ docker run -d --name sensu-backend \
-p 2380:2380 -p 3000:3000 -p 8080:8080 -p 8081:8081 \
sensu/sensu:master sensu-backend start

$ docker run -d --name sensu-agent --link sensu-backend \
sensu/sensu:master sensu-agent start \
--backend-url ws://sensu-backend:8081 \
--subscriptions workstation,docker
```

Sensu

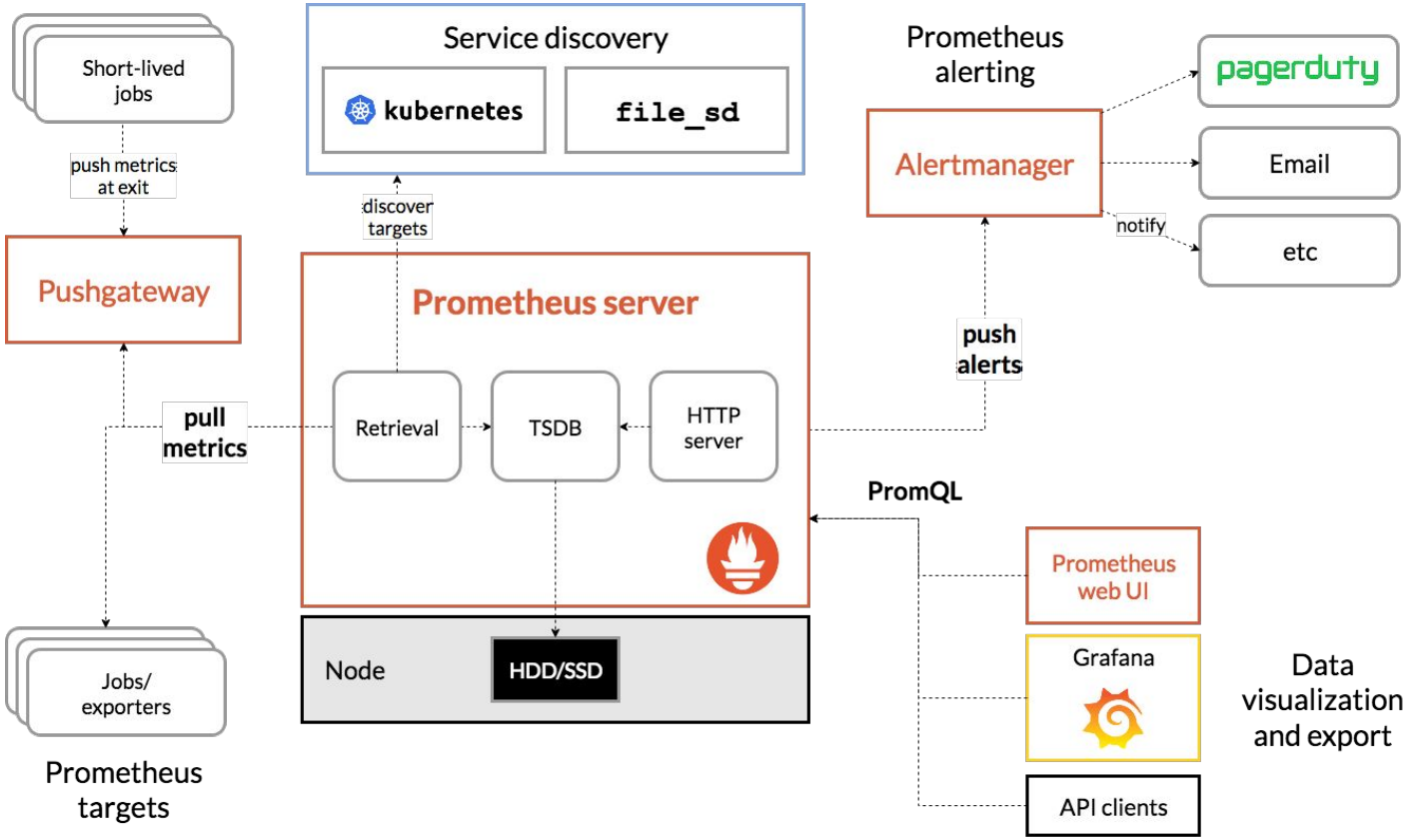
Prometheus

Prometheus features

- Multi-dimensional time series model
- Pull model (HTTP scraping)
  - Optional push model (via a push gateway)
- Exporters
- Node discovery
  - Static
  - Service discovery integration

Prometheus

Prometheus architecture



Data model


```
metric_name [
  "{" label_name "=" `"` label_value `"` { "," label_name "=" `"`
label_value `"` } [ "," ] "}"
] value [ timestamp ]
```

- Counter
- Gauge
- Histogram
- Summary

Prometheus

Data model


```
# HELP http_requests_total The total number of HTTP requests.
# TYPE http_requests_total counter
http_requests_total{method="post",code="200"} 1027 1395066363000
http_requests_total{method="post",code="400"}    3 1395066363000


# A histogram, which has a pretty complex representation in the text format:
# HELP http_request_duration_seconds A histogram of the request duration.
# TYPE http_request_duration_seconds histogram
http_request_duration_seconds_bucket{le="0.05"} 24054
http_request_duration_seconds_bucket{le="0.1"} 33444
http_request_duration_seconds_bucket{le="0.2"} 100392
http_request_duration_seconds_bucket{le="0.5"} 129389
http_request_duration_seconds_bucket{le="1"} 133988
http_request_duration_seconds_bucket{le="+Inf"} 144320
http_request_duration_seconds_sum 53423
http_request_duration_seconds_count 144320
```

Prometheus

Queries (PromQL)


```
http_requests_total{environment=~"staging|development",method!="GET"}

http_requests_total offset 5m

http_requests_total{job="prometheus"}[5m]

rate(http_requests_total{job="api-server"}[5m])

topk(5, http_requests_total)
```

Prometheus

Configuration


```
global:
  scrape_interval:     15s
  evaluation_interval: 15s

rule_files:
   - "alert.rules"

scrape_configs:
  - job_name: prometheus
    static_configs:
      - targets: ['localhost:9090']
```

Prometheus

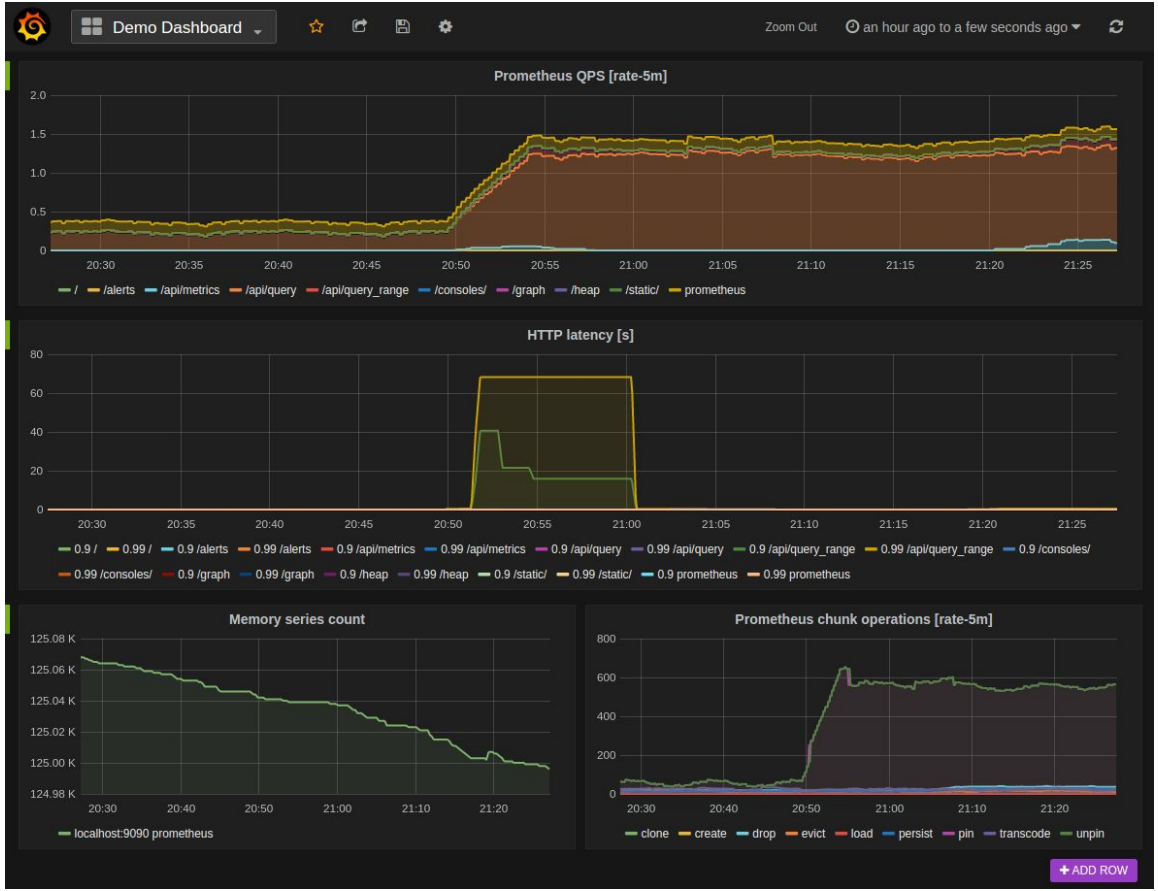
Alerting


```
groups:
- name: example
  rules:
  - alert: HighErrorRate
    expr: job:request_latency_seconds:mean5m{job="myjob"} > 0.5
    for: 10m
    labels:
      severity: page
    annotations:
      summary: High request latency
```

Prometheus

Grafana

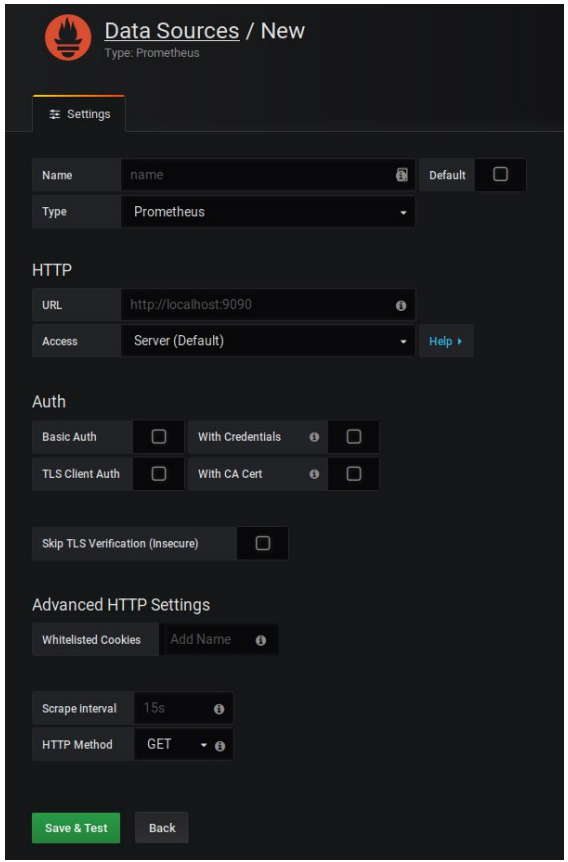
$> t3chfest

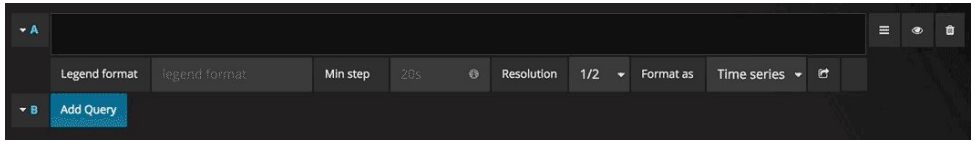


Prometheus

Grafana - Add Prometheus as Data Source

$> t3chfest





Prometheus

Deploying Prometheus


```
docker run -p 9090:9090 -v
/tmp/prometheus.yml:/etc/prometheus/prometheus.yml \
       prom/prometheus

docker run -d --name=grafana --net="host" \
grafana/grafana

docker run -d \
  --net="host" \
  --pid="host" \
  -v "/:/host:ro,rslave" \
  quay.io/prometheus/node-exporter \
  --path.rootfs /host
```

Prometheus

Each system in a sentence

elastic is for logs

influxdata: time series on steroids

Nagios upgraded

Sensu-go: The beauty of simplicity

From 0 to Grafana in 10 minutes

Happy hacking!

Alejandro Guirao

@lekum

lekum.org

CARTO