Alerting with Time Series

Fabian Reinartz, CoreOS

github.com/fabxc

@fabxc
Stream of <timestamp, value> pairs associated with an identifier

http_requests_total{job="nginx",instance="1.2.3.4:80",path="/status",status="200"}

1348 @ 1480502384
1899 @ 1480502389
2023 @ 1480502394

http_requests_total{job="nginx",instance="1.2.3.1:80",path="/settings",status="201"}
http_requests_total{job="nginx",instance="1.2.3.5:80",path="/",status="500"}
...

Time Series
Stream of <timestamp, value> pairs associated with an identifier

\[
\text{sum by(path, status) (rate(http_requests_total\{job="nginx\}\[5m\]))}
\]

{path="/status", status="200"} 32.13  @ 1480502384
{path="/status", status="500"} 19.133 @ 1480502394
{path="/profile", status="200"} 44.52  @ 1480502389
A lot of traffic to monitor

Monitoring traffic should not be proportional to user traffic
A lot of targets to monitor

A single host can run hundreds of machines/procs/containers/...
Targets constantly change

Deployments, scaling up, scaling down, rolling-updates
Need a fleet-wide view

What’s my 99th percentile request latency across all frontends?
Drill-down for investigation

Which pod/node/... has turned unhealthy? How and why?
Monitor all levels, with the same system

Query and correlate metrics across the stack
Translate that to

Meaningful Alerting
If you are actually monitoring at scale, *something will always correlate.*

Huge efforts to eliminate huge number of false positives.

Huge chance to introduce false negatives.
Prometheus Alerts

![Diagram]

- **current state**
- **desired state**
- **alerts**
Symptom-based pages

Urgent issues – Does it hurt your user?
Latency

Four Golden Signals
Traffic

Four Golden Signals
Errors

Four Golden Signals
Cause-based warnings

Helpful context, non-urgent problems

user

system

dependency

dependency

dependency

dependency
Saturation / Capacity

Four Golden Signals

user

system

dependency

dependency

dependency

dependency
Prometheus Alerts

**Alert Syntax**

```
ALERT <alert name>
  IF <PromQL vector expression>
  FOR <duration>
  LABELS { ... }
  ANNOTATIONS { ... }
```

Each result entry is one alert:

```
<elem1>   <val1>
<elem2>   <val2>
<elem3>   <val3>
...
```
etcd_has_leader{job="etcd", instance="A"} 0
etcd_has_leader{job="etcd", instance="B"} 0
etcd_has_leader{job="etcd", instance="C"} 1
Prometheus Alerts

```
ALERT EtcdNoLeader
    IF etcd_has_leader == 0
    FOR 1m
    LABELS {
        severity="page"
    }
    {
        job="etcd", instance="A"
    } 0.0
    {
        job="etcd", instance="B"
    } 0.0
    {
        job="etcd", alertname="EtcdNoLeader", severity="page", instance="A"
    }
    {
        job="etcd", alertname="EtcdNoLeader", severity="page", instance="B"
    }
```
requests_total{instance="web-1", path="/index", method="GET"} 8913435
requests_total{instance="web-1", path="/index", method="POST"} 34845
requests_total{instance="web-3", path="/api/profile", method="GET"} 654118
requests_total{instance="web-2", path="/api/profile", method="GET"} 774540

request_errors_total{instance="web-1", path="/index", method="GET"} 84513
request_errors_total{instance="web-1", path="/index", method="POST"} 434
request_errors_total{instance="web-3", path="/api/profile", method="GET"} 6562
request_errors_total{instance="web-2", path="/api/profile", method="GET"} 3571
ALERT HighErrorRate

IF sum(rate(request_errors_total[5m])) > 500

{}  534
Alert HighErrorRate

IF sum(rate(request_errors_total[5m])) > 500

Absolute threshold alerting rule needs constant tuning as traffic changes
ALERT HighErrorRate

IF sum(rate(request_errors_total[5m])) > 500

d{} 534

traffic changes over days
ALERT HighErrorRate

IF sum(rate(request_errors_total[5m])) > 500

traffic changes over months
ALERT HighErrorRate

    IF sum(rate(request_errors_total[5m])) > 500

traffic when you release awesome feature X
ALERT HighErrorRate

IF sum(rate(request_errors_total[5m])) / sum(rate(requests_total[5m])) > 0.01

{}  1.8354
Alert HighErrorRate

If \( \frac{\text{sum(rate(request_errors_total[5m]))}}{\text{sum(rate(requests_total[5m]))}} > 0.01 \)

\[
\begin{align*}
\{ & 1.8354
\end{align*}
\]
ALERT HighErrorRate

IF sum(rate(request_errors_total[5m])) /
    sum(rate(requests_total[5m])) > 0.01

{} 1.8354

No dimensionality in result
loss of detail, signal cancelation
ALERT HighErrorRate

IF \[ \frac{\text{sum(rate(request\_errors\_total[5m]))}}{\text{sum(rate(requests\_total[5m]))}} > 0.01 \]
**Alert** HighErrorRate

- IF \( \frac{\text{sum by(instance, path) (rate(request_errors_total[5m])}}{\text{sum by(instance, path) (rate(requests_total[5m]))}} > 0.01 \)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Path</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>web-2</td>
<td>/api/comments</td>
<td>0.02435</td>
</tr>
<tr>
<td>web-1</td>
<td>/api/comments</td>
<td>0.01055</td>
</tr>
<tr>
<td>web-2</td>
<td>/api/profile</td>
<td>0.34124</td>
</tr>
</tbody>
</table>
ALERT HighErrorRate

IF sum by(instance, path) (rate(request_errors_total[5m])) / 
    sum by(instance, path) (rate(requests_total[5m])) > 0.01

{instance="web-2", path="/api/v1/comments"} 0.022435
...

Wrong dimensions
aggregates away dimensions of fault-tolerance
ALERT HighErrorRate

IF \[
\frac{\text{sum by(instance, path) (rate(request_errors_total[5m]))}}{\text{sum by(instance, path) (rate(requests_total[5m]))}} > 0.01
\]

\{instance="web-2", path="/api/v1/comments"\} 0.02435

...
ALERT HighErrorRate

IF sum without(instance) (rate(request_errors_total[5m])) / sum without(instance) (rate(requests_total[5m])) > 0.01

{method="GET", path="/api/v1/comments"} 0.02435
{method="POST", path="/api/v1/comments"} 0.015
{method="POST", path="/api/v1/profile"} 0.34124
ALERT DiskWillFillIn4Hours

IF predict_linear(node_filesystem_free{job='node'}[1h], 4*3600) < 0 FOR 5m

...
ALERT DiskWillFillIn4Hours

IF predict_linear(node_filesystem_free{job='node'}[1h], 4*3600) < 0
FOR 5m

ANNOTATIONS {
  summary = "device filling up",
  description = "{{$labels.device}} mounted on {{$labels.mountpoint}} on
  {{$labels.instance}} will fill up within 4 hours."
}
Alertmanager

Aggregate, deduplicate, and route alerts
Prometheus

Targets

Service Discovery
(Kubernetes, AWS, Consul, custom...)

Prometheus

Alertmanager

Email, Slack, PagerDuty, OpsGenie, ...
<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>04:11</td>
<td>hey, HighLatency,</td>
<td>service=&quot;X&quot;, zone=&quot;eu-west&quot;, path=/user/profile, method=GET</td>
</tr>
<tr>
<td>04:11</td>
<td>hey, HighLatency,</td>
<td>service=&quot;X&quot;, zone=&quot;eu-west&quot;, path=/user/settings, method=GET</td>
</tr>
<tr>
<td>04:11</td>
<td>hey, HighLatency,</td>
<td>service=&quot;X&quot;, zone=&quot;eu-west&quot;, path=/user/settings, method=GET</td>
</tr>
<tr>
<td>04:11</td>
<td>hey, HighErrorRate,</td>
<td>service=&quot;X&quot;, zone=&quot;eu-west&quot;, path=/user/settings, method=POST</td>
</tr>
<tr>
<td>04:12</td>
<td>hey, HighErrorRate,</td>
<td>service=&quot;X&quot;, zone=&quot;eu-west&quot;, path=/user/profile, method=GET</td>
</tr>
<tr>
<td>04:13</td>
<td>hey, HighLatency,</td>
<td>service=&quot;X&quot;, zone=&quot;eu-west&quot;, path=/index, method=POST</td>
</tr>
<tr>
<td>04:13</td>
<td>CacheServerSlow,</td>
<td>service=&quot;X&quot;, zone=&quot;eu-west&quot;, path=/user/profile, method=POST</td>
</tr>
<tr>
<td>04:15</td>
<td>hey, HighErrorRate,</td>
<td>service=&quot;X&quot;, zone=&quot;eu-west&quot;, path=/comments, method=GET</td>
</tr>
<tr>
<td>04:15</td>
<td>hey, HighErrorRate,</td>
<td>service=&quot;X&quot;, zone=&quot;eu-west&quot;, path=/user/profile, method=POST</td>
</tr>
</tbody>
</table>
You have 15 alerts for **Service X** in **zone eu-west**

- 3x HighLatency
- 10x HighErrorRate
- 2x CacheServerSlow

Individual alerts: ...
Inhibition

- {alertname="DatacenterOnFire", severity="page", zone="eu-west"}
  - if active, mute everything else in same zone
- {alertname="LatencyHigh", severity="page", ..., zone="eu-west"}
- ...
- {alertname="ErrorsHigh", severity="page", ..., zone="eu-west"}
- {alertname="ServiceDown", severity="page", ..., zone="eu-west"}
Anomaly Detection
Practical Example 1

```plaintext
job:requests:rate5m = sum by(job) (rate(requests_total[5m]))

job:requests:holt_winters_rate1h = holt_winters(
  job:requests:rate5m[1h], 0.6, 0.4
)```
Practical Example 1

```
ALERT AbnormalTraffic
IF abs(
    job:requests:rate5m - job:requests:holt_winters_rate1h offset 7d
) > 0.2 * job:request_rate:holt_winters_rate1h offset 7d
FOR 10m
...
```
Practical Example 2

instance:latency_seconds:mean5m > on (job) group_left()
    (avg by (job)(instance:latency_seconds:mean5m) + on (job)
       2 * stddev by (job)(instance:latency_seconds:mean5m))
Practical Example 2

```sql

(instance:latency_seconds:mean5m > on (job) group_left()
 ( avg by (job)(instance:latency_seconds:mean5m) + on (job) 2 * stddev by (job)(instance:latency_seconds:mean5m) ) )

> on (job) group_left()
1.2 * avg by (job)(instance:latency_seconds:mean5m)
```
(instance:latency_seconds:mean5m > on (job) group_left()
  (avg by (job)(instance:latency_seconds:mean5m) + on (job)
    2 * stddev by (job)(instance:latency_seconds:mean5m))
) > on (job) group_left()
  1.2 * avg by (job)(instance:latency_seconds:mean5m)

and on (job)
  avg by (job)(instance:latency_seconds_count:rate5m) > 1
Self Healing
Conclusion

- Symptom-based pages + cause based warnings provide good coverage and insight into service availability
  - Design alerts that are adaptive to change, preserve as many dimensions as possible, aggregate away dimensions of fault tolerance
  - Use linear prediction for capacity planning and saturation detection
- Advanced alerting expressions allow for well-scoped and practical anomaly detection

- Raw alerts are not meant for human consumption
- The Alertmanager aggregates, silences, and routes groups of alerts as meaningful notifications
Join us!
careers: coreos.com/careers (now in Berlin!)