@snehainguva

prometheus everything,
observing **kubernetes** in the
cloud

about me

software engineer @DigitalOcean
former **delivery,** currently **observability**
kubernetes, prometheus

Some **stats**

15 kubernetes clusters

12 data centers

300+ production applications

+

2 promethei +
1 alertmanager per cluster

1.5 million+ timeseries

99218 samples/sec
(note: data-center wide scraping is at **550k** samples/sec)

the plan:

- the **pre-kubernetes** days
- **kubernetes** at DigitalOcean (aka **docc**)
- **prometheus** + **alertmanager** and **kubernetes**
- alerting in action: examples
- potential pitfalls
- next steps

pre-kubernetes:

1. service owners write an application
2. provision a server with chef or ansible
3. use a CI/CD pipeline, bash scripts, or other tools

   to deploy and update application on a VM

pre-kubernetes:

4. use nagios + various plugins to monitor
5. use collectd + application metrics + statsd + graphite
6. push data to openTSDB

pre-kubernetes:

longer to provision host than write actual service

blackbox monitoring **NOT** insightful

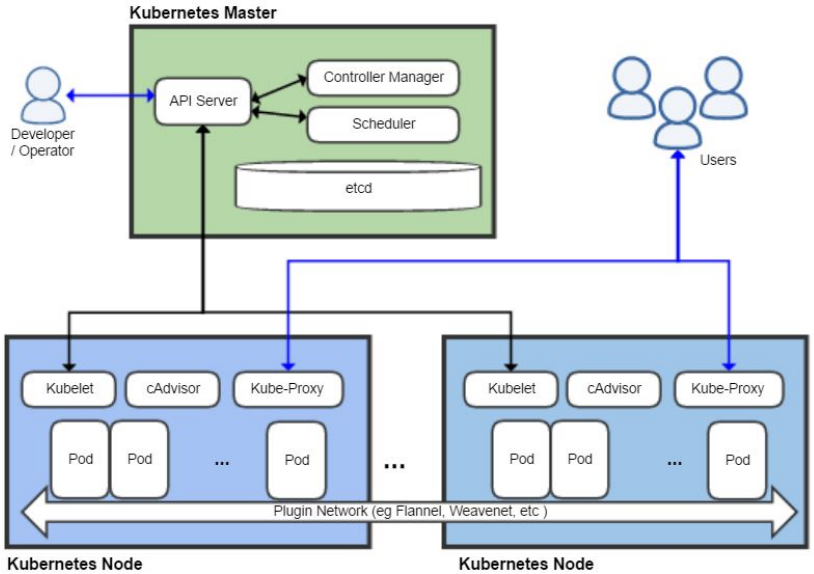
whitebox monitoring services **NOT** easily **queryable**

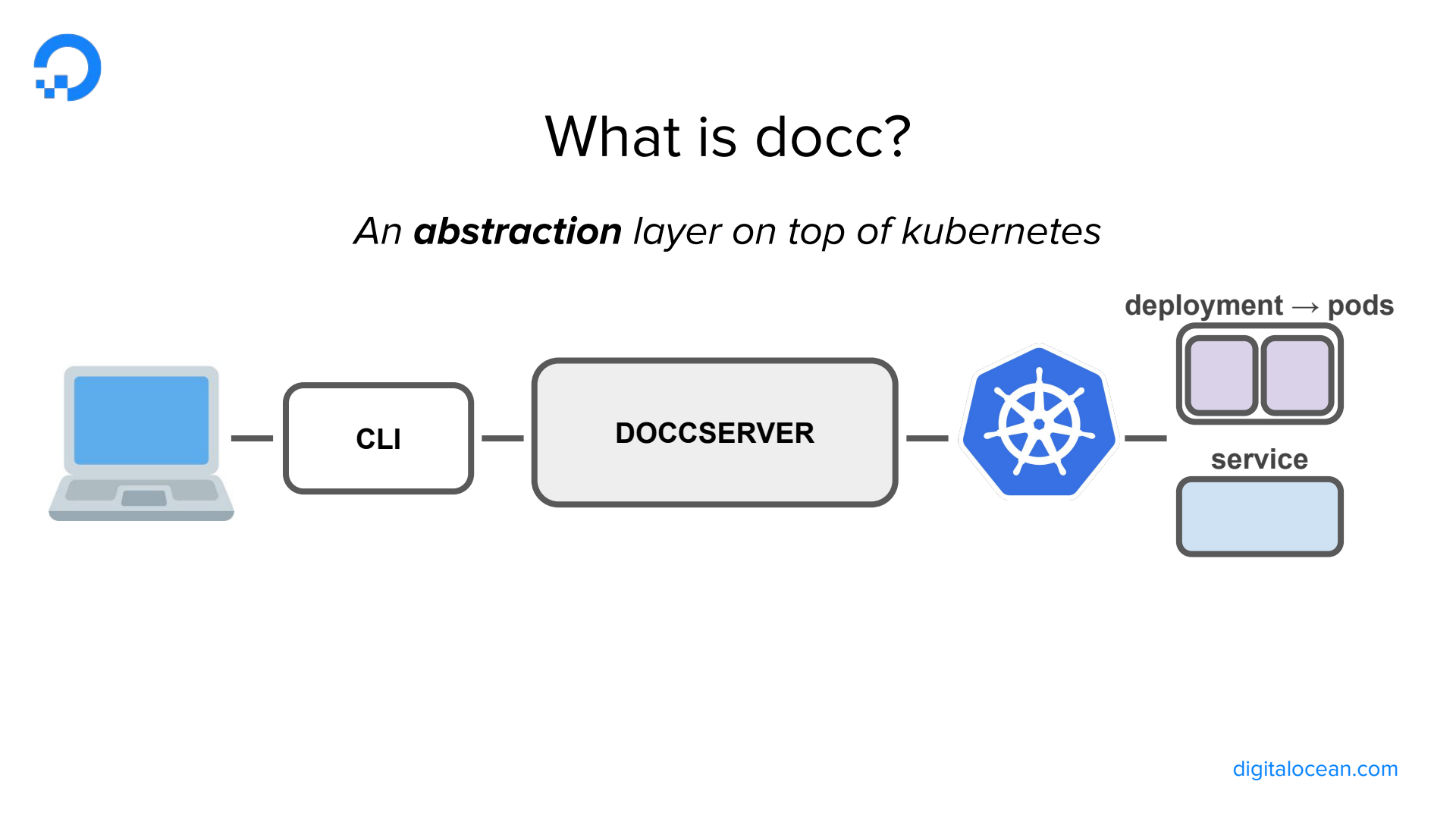
docc: **D**igital **O**cean **C**ommand **C**enter

A tool for deploying ***containerized****,*
stateless *applications*

What is kubernetes?

*Container **orchestration tool** from Google*





What is docc?

*An **abstraction** layer on top of kubernetes*

post-docc:

1. service owners write an application
2. service owner dockerizes application
3. describe application in json manifest file
4. **deploy!**

post-docc:

deployments and updates take **minutes**, not hours

view running **applications**

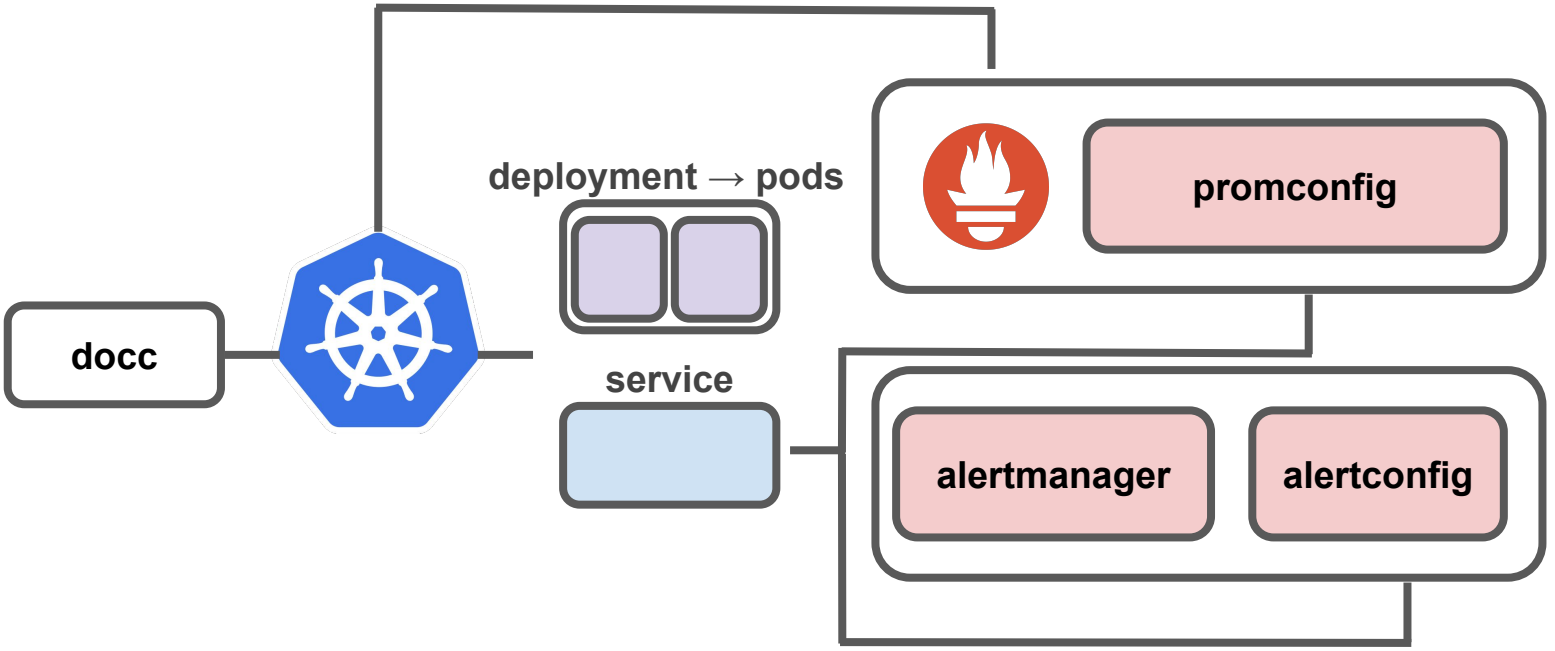
get application **logs**

easily **scale**, **update**, or **restart** applications

But what about **monitoring?**

Let’s use

prometheus + **alertmanager**


deployment → pods
promconfig
docc
service
alertmanager
alertconfig

1

instrument your application

use prometheus golang client

expose **metrics endpoint**

2

specify **metrics, ports, alerts** in your manifest file

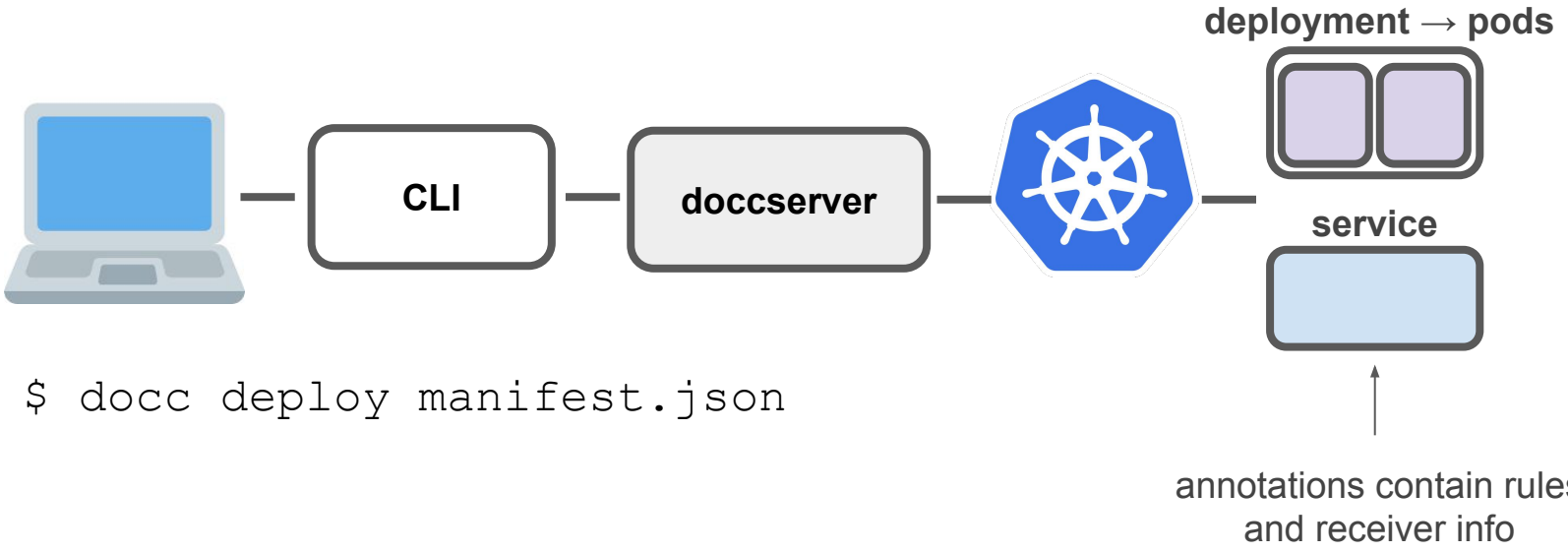
Which **metrics endpoint** should be scraped?

Which container **port** needs to be exposed?

Specify **alerting rule**, **duration** interval, and **channel**.

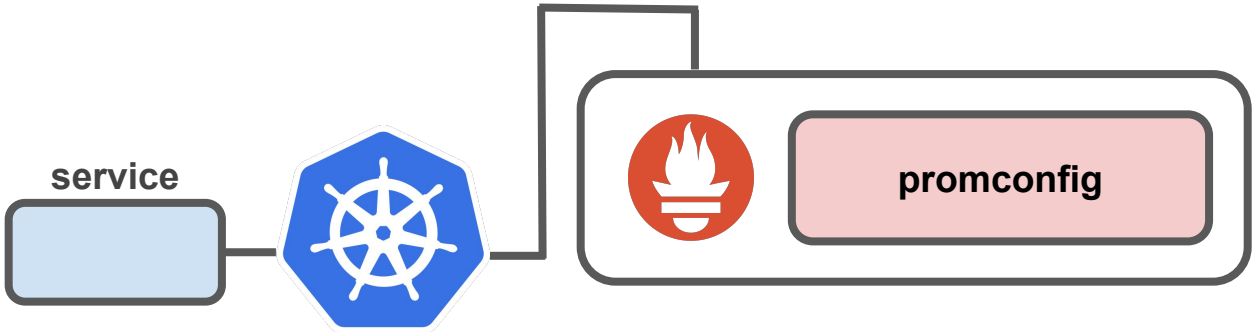
3

use **docc CLI** to deploy your application



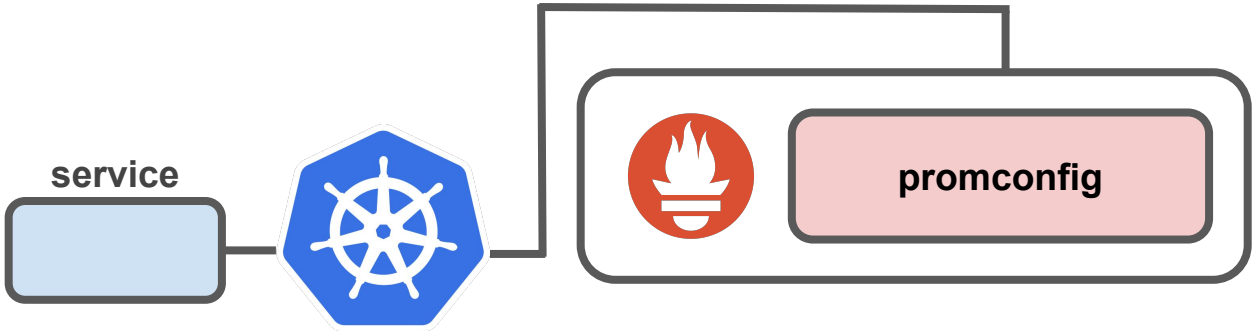
4

prometheus talks to the **kubernetes api** and grabs the metrics endpoint and port information



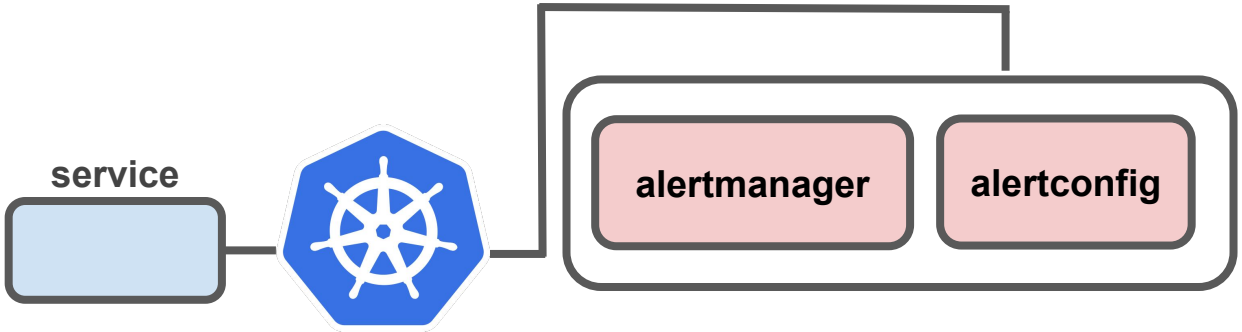
5

promconfig grabs alert information and rewrites prometheus rules file



6

alertconfig grabs alert routes and rewrites alertmanager configuration file



What should we **monitor**?

4 Golden Signals

request-based system metrics

latency

traffic

error

saturation

Request

Errors

Duration

Brendan Gregg's USE-ful metrics

"Solves 80% of server issues with 5% of the effort."

Utilization

Saturation

Error

prom metrics types

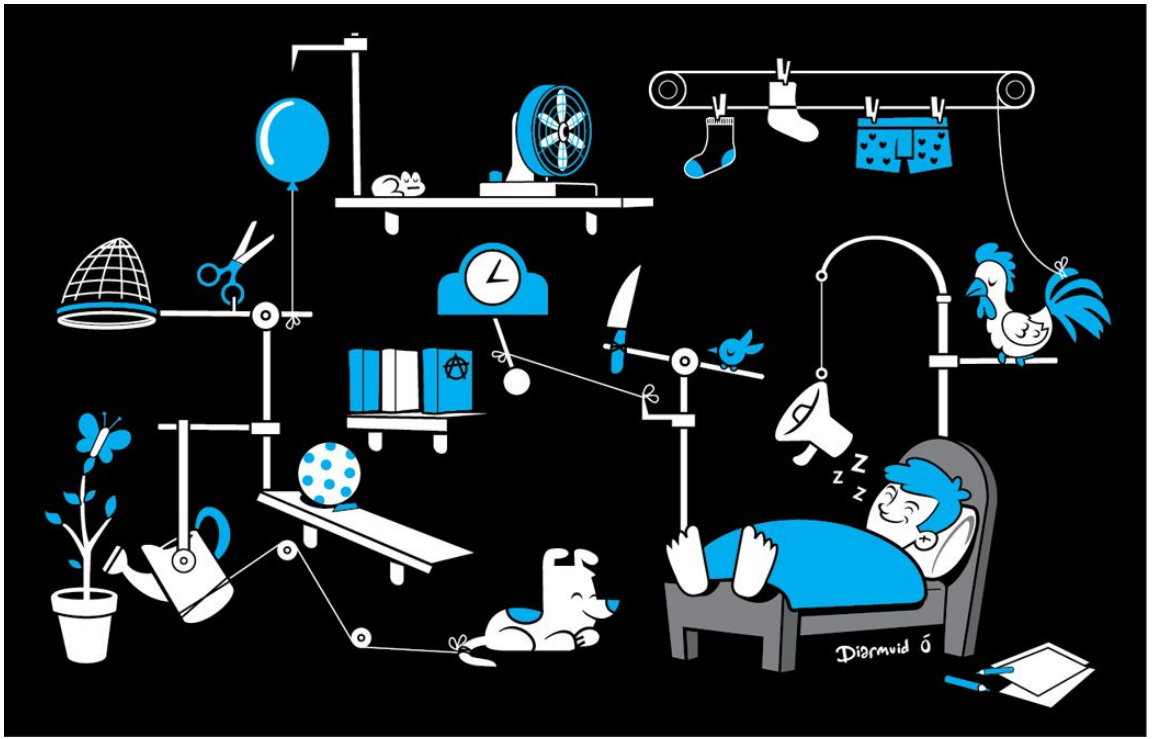
counters: *cumulative, increasing metric*

gauges: *single metric that goes up or down*

histograms: *samples and buckets observations*

summaries: *samples observations, specify quantile*

Putting it all together...



service metric: traffic

how much demand is placed on the system

loadbalancer backend traffic

fxn: rate() and sum()

metric type: counter

```
sum(rate(haproxy_backend_bytes_out_total{
    kubernetes_name="loadbalancer",
    backend="tls_default_neptune_nyc3_internal_digitalocean_com"}
[1m])) BY (backend)
```

labels

cluster metric: utilization

average time resource is busy servicing work

cluster CPU utilization

fxn: sum() and rate()

metric type: counter

```
(sum(rate(container_cpu_usage_seconds_total{id="/"}[5m]))
/ sum(machine_cpu_cores))
```

How should we **alert**?

Threshold alerts

*Do any of the aforementioned metrics exceed a **lower** or **upper bound**?*

Threshold alerts

Are more than 80% of cluster CPU cores being utilized?

```
(sum(rate(container_cpu_usage_seconds_total{id="/"}[5m]))
/ sum(machine_cpu_cores))* 100 > 80
```

State-based alerts

*Is there a divergence between **expected state** and **actual state** of a service?*

State-based alerts

Is my service up and/or scrape-able?

```
absent(up{kubernetes_name="doccserver"}) or
sum(up{kubernetes_name="doccserver"}) == 0
```

Common **pitfalls**

Pitfall #1: **Alerting fatigue**

Sneha Inguva 3:58 PM
okay so i think i may have confused myself on how pagerduty is deduplicating stuff so gonna page myself a bunch for our testservice

Sneha Inguva 6:15 PM
Crap! Did i just miss a pagerduty page? What happened

Solution: **Slack** and/or **Pagerduty**

send only the most **urgent, production alerts** to pagerduty

try out different **promQL queries** to have **less spikey** metrics

Pitfall #2: **Who owns what?**

Sneha Inguva 8:53 PM

well that was quite the spike

8:53 ☆ what is neptune, actually, btw?

Mac Browning 8:56 PM

sweet

wasnt sure, just did a `docc show` to find the maintainer

Solution: opinionated **manifest** file

services owner **must** include **maintainer** information

alerts themselves include **descriptions** and **summaries** with

several labels

alerts must include **team-specific** receivers

Pitfall #3: **Meta-monitoring**

Mitchell Anicas 10:49 AM
is frog's prometheus down?

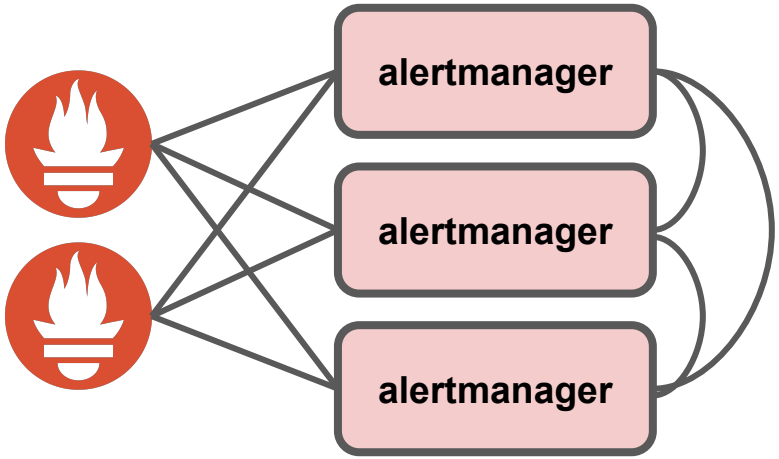
Joonas Bergius 10:50 AM
let's see

Sneha Inguva 10:51 AM
hey @manicas taking a look!

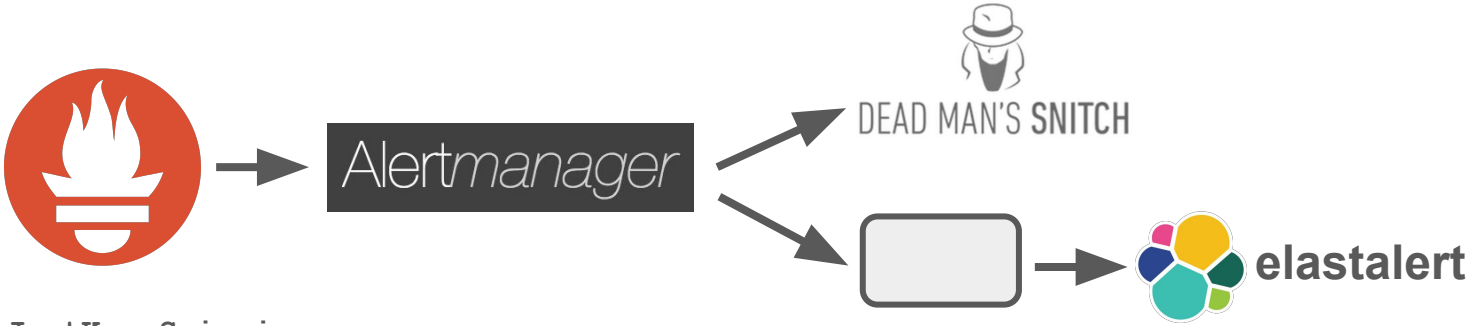
Tommy Murphy 10:58 AM
ha and now im taking over

yeah it sure does look to be down!

Solution: **Duplicate promethei and HA alertmanager**



Solution: **Deadman's switch**




```
ALERT JustKeepSwimming
  IF vector(1)
```

#1: **Automated alerts**

utilize user-defined memory and cpu limits for **threshold alerts**

automatic **state-based** alerts

#2: **Leverage metrics for autopilot**

user trusts in our custom **controllers** and **schedulers**

collect metrics and **build model** about resource usage over time

accordingly adjust **limits** and **alerts**

#3: **Leverage metrics for autoscaling**

services based on resource usage, # connections, etc.

loadbalancers based on # of frontend and backend connections

of **worker nodes** based on memory and cpu capacity metrics

a brave new world of **container orchestration**

prometheus + **alertmanager** are awesome!

extensibility

thanks!

- @snehainguva
- The best prometheus tutorials you will ever read, Julius Volz
- Actual Prometheus Website
- Kubernetes Project