

Kubernetes monitoring

Why it is difficult and how to improve it



Aliaksandr Valialkin

VictoriaMetrics founder and core developer. Go contributor and author of popular libraries fasthttp, fastcache, quicktemplate



Time Series Database & Monitoring Solution

- Open source
- Simple setup & operation
- Cost efficient
- Highly scalable
- Cloud ready

Cloud-native impact on observability data volumes

Observability data is growing 2-3x faster than application data

Observability data increase in scale

Business increase in scale



Source: <https://chronosphere.io/learn/new-study-uncovers-top-observability-concerns-in-2022>



Elan 🙄

@ElanHasson



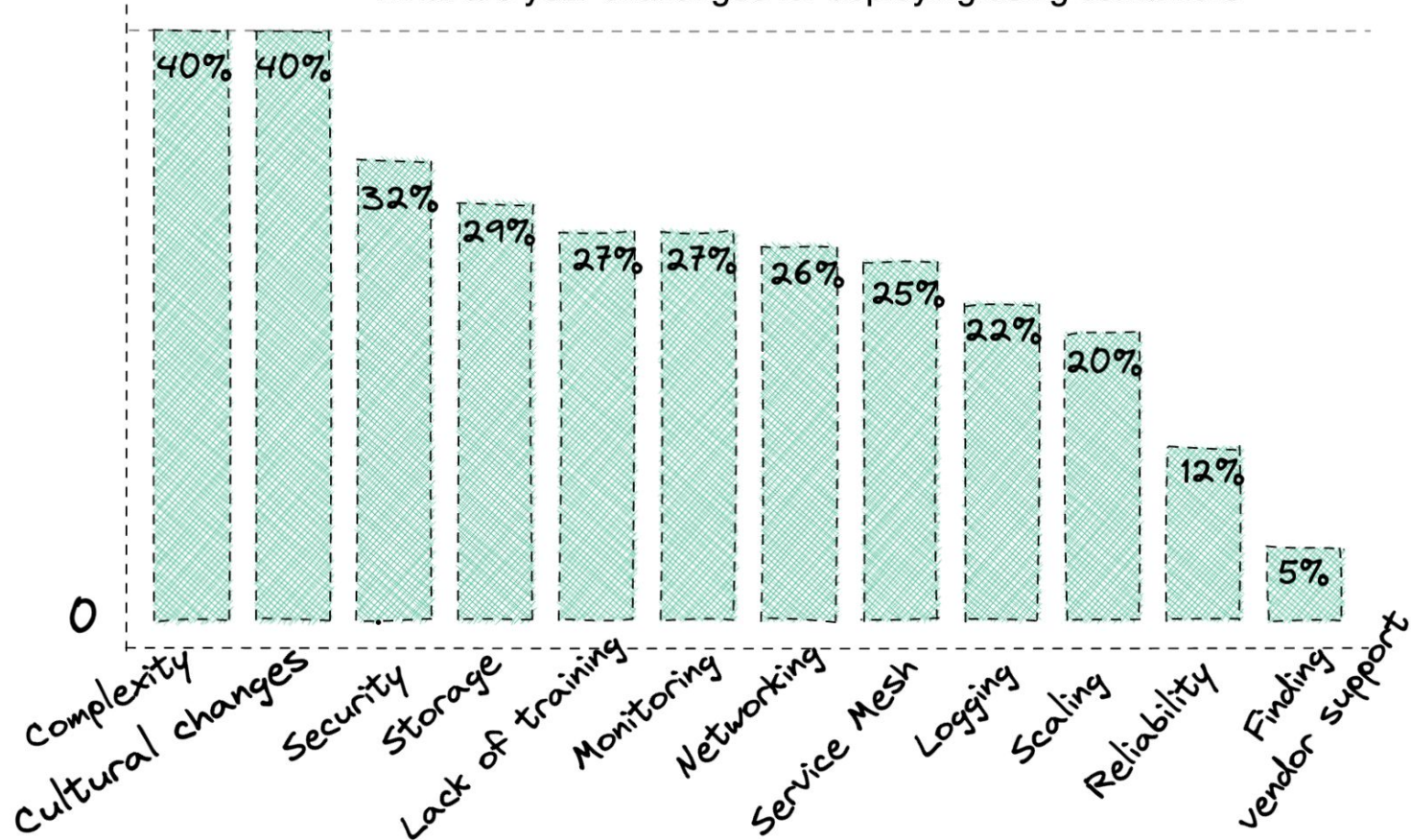
Replying to [@d_crosario](#) and [@GergelyOrosz](#)

Paying more for logging/metrics/tracing doesn't equate to a positive user experience.

Consider how much data can be generated and shipped. \$\$\$.

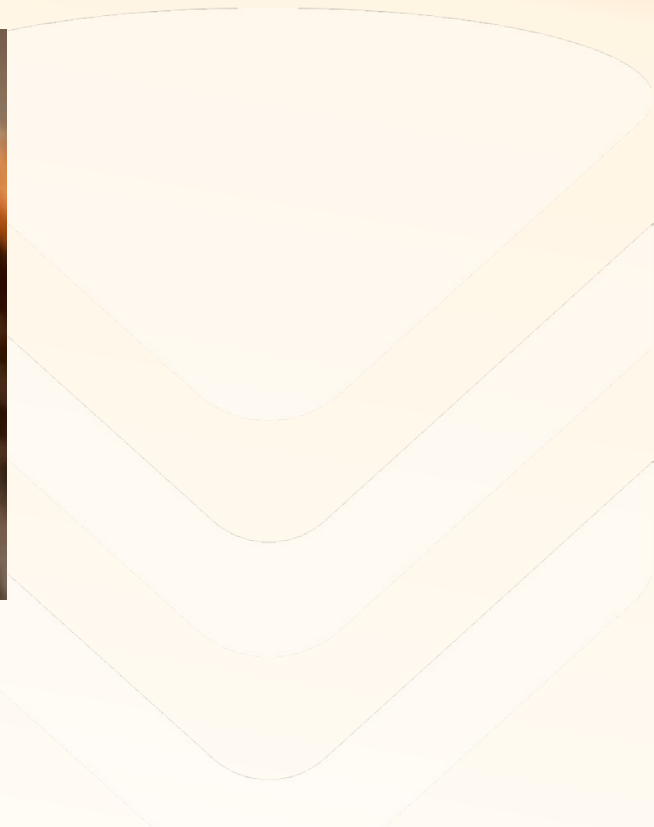
You still need good people to turn data into action.

What are your challenges for deploying/using containers



Source: CNCF Survey report 2020 - https://www.cncf.io/wp-content/uploads/2020/11/CNCF_Survey_Report_2020.pdf

Why Kubernetes monitoring is so challenging?



Kubernetes metrics

- K8s exposes big amounts of metrics

<https://kubernetes.io/docs/concepts/cluster-administration/system-metrics/>

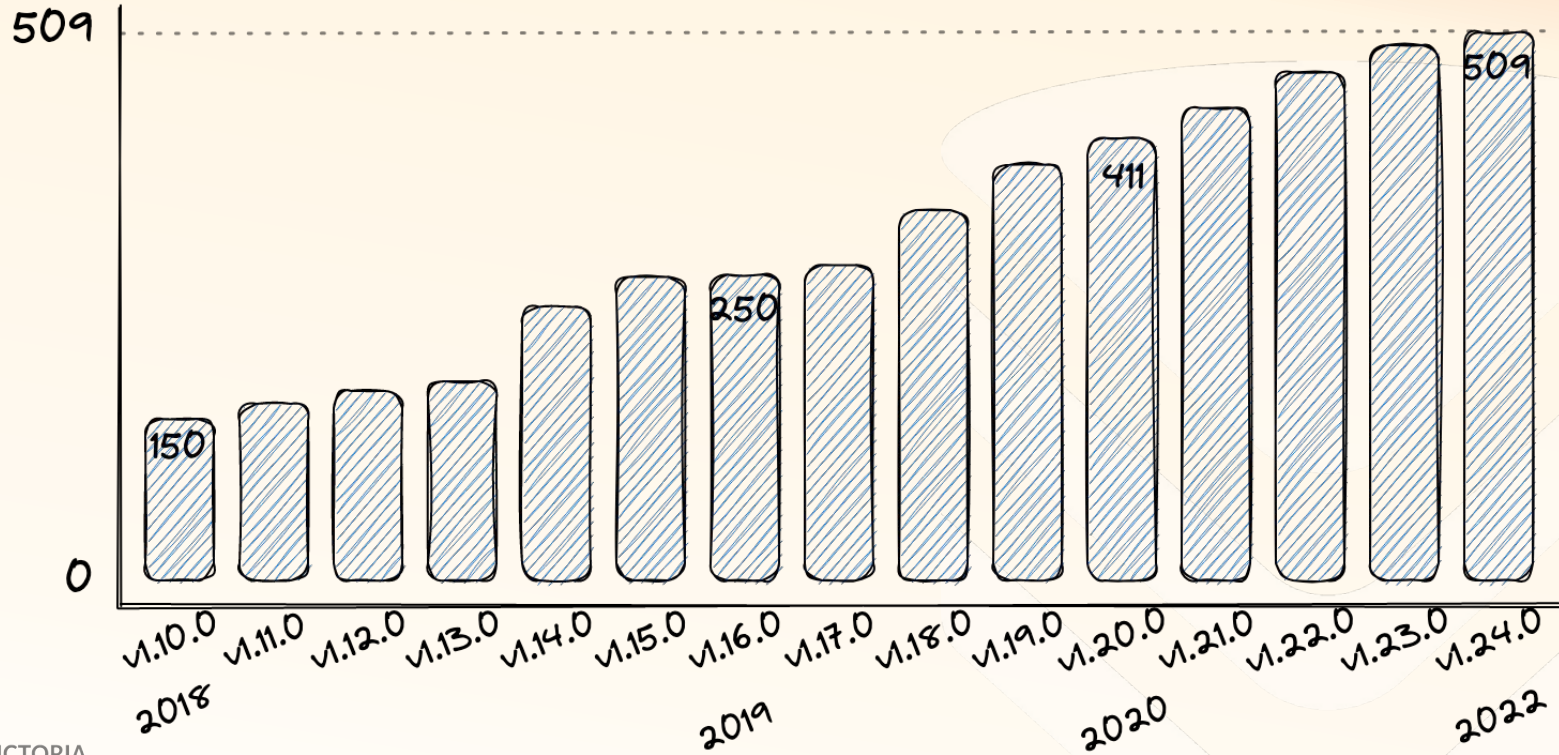
Kubernetes metrics

- K8s exposes big amounts of metrics

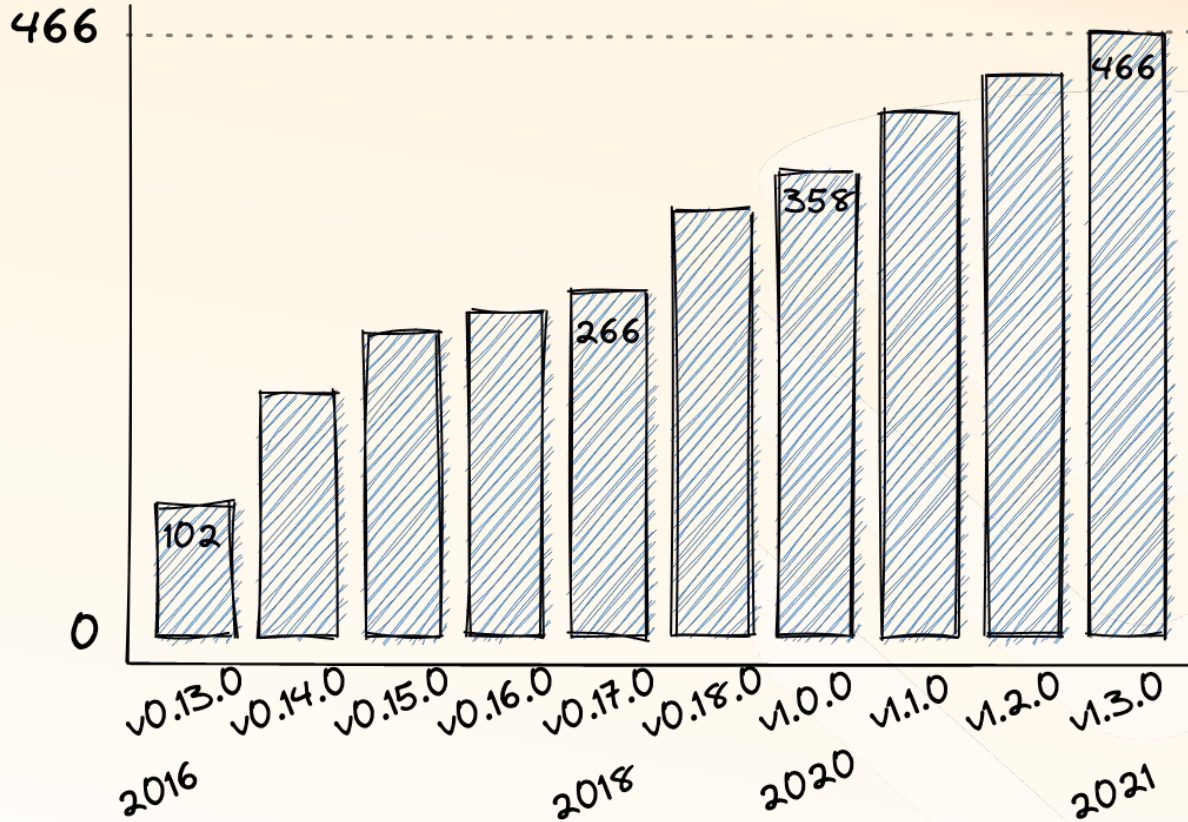
<https://kubernetes.io/docs/concepts/cluster-administration/system-metrics/>

- The number of exposed metrics grows over time

Metric names for K8s - 3.4x growth



Metric names for node_exporter - 4.5x growth



Millions of metrics

For Kubernetes version 1.24.0 every node exports at least 2459 series
(not counting application metrics)

node * (node_exporter + kubelet&cadvisor)

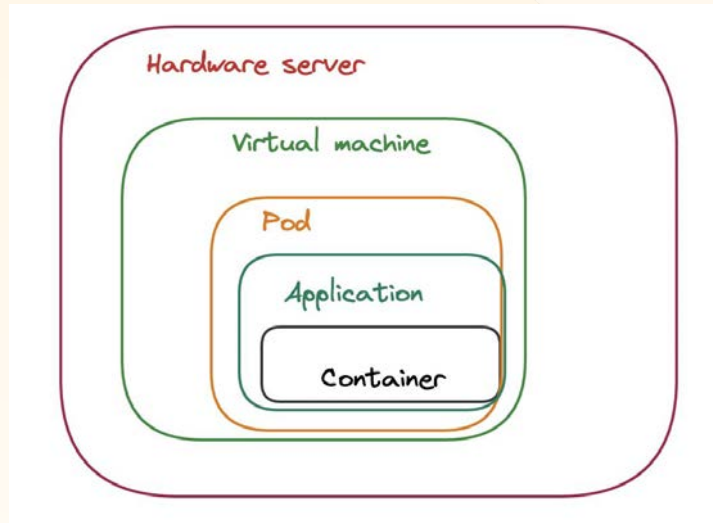
minimal ~ (663 + 1796) = 2459

realistic ~ (1576 + 2530) = 4106

** realistic data is an average across different clusters we were able get info from*

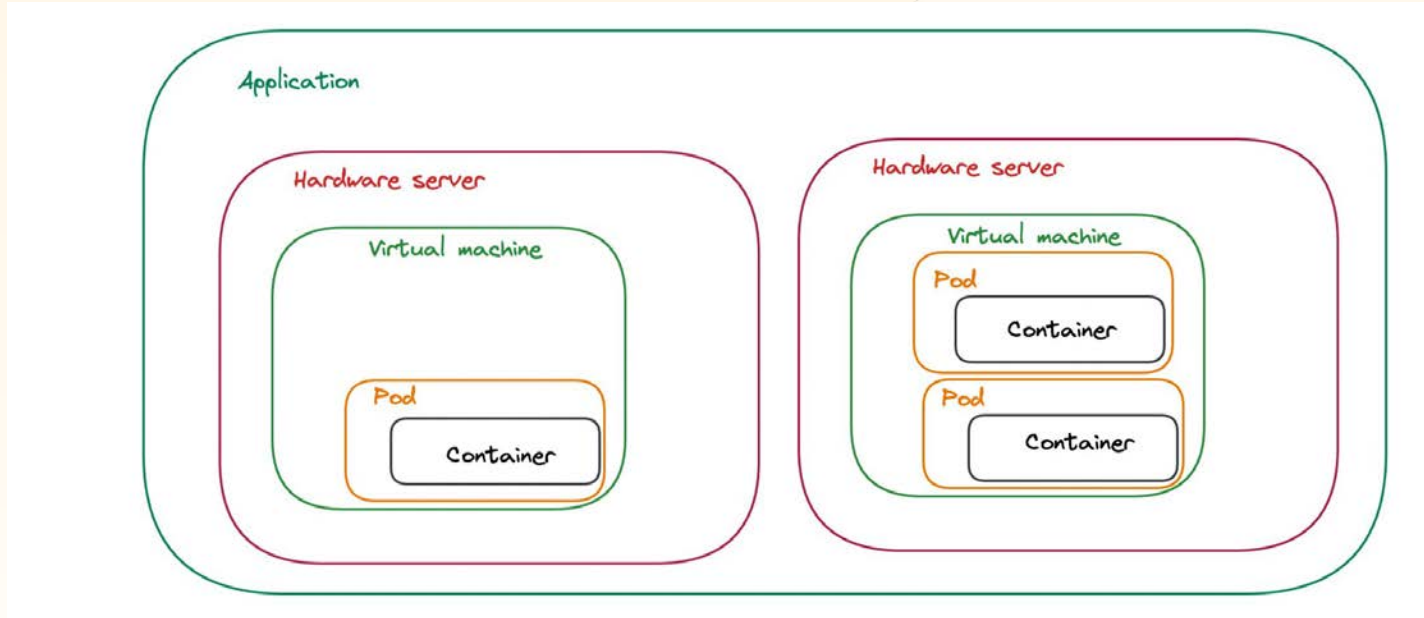
Millions of metrics

- Too many layers introduce extra complexity and cognitive load



Millions of metrics

- The number of metrics increases with the number of K8s containers



Millions of metrics

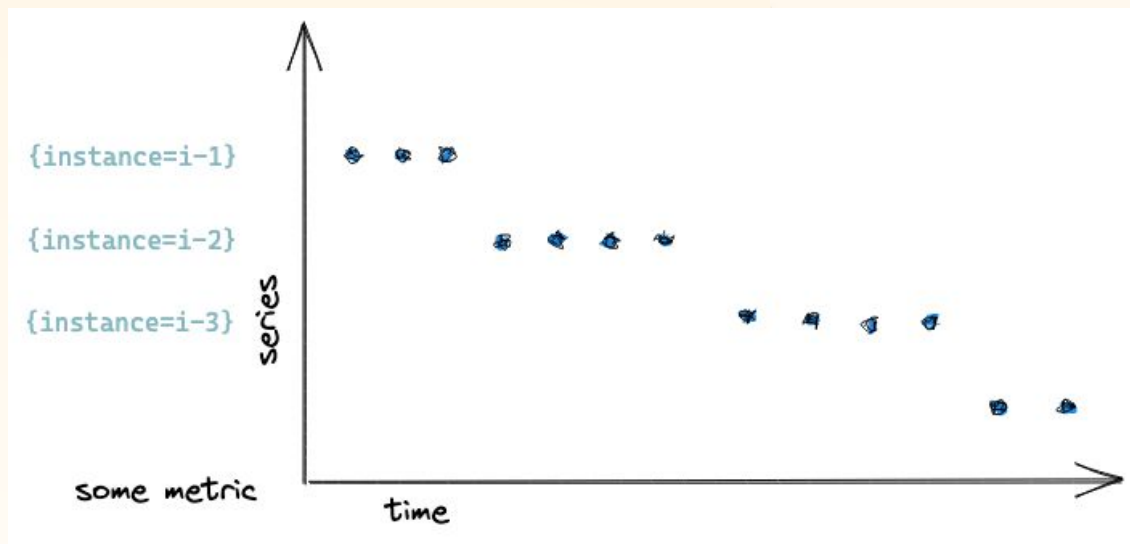
A simple Deployment of 1 container with 3 replicas adds 629 new metric series generated by cadvisor

Not counting metrics exposed by application

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4  | name: nginx-deployment
5  spec:
6  | selector:
7  | | matchLabels:
8  | | | app: nginx
9  | replicas: 3
10 | template:
11 | | metadata:
12 | | | labels:
13 | | | | app: nginx
14 | | spec:
15 | | | containers:
16 | | | - name: nginx
17 | | | | image: nginx:1.14.2
18 | | | | ports:
19 | | | | - containerPort: 80
20 | ---
21 apiVersion: v1
22 kind: Service
23 metadata:
24 | name: hello-world
25 spec:
26 | selector:
27 | | app: nginx
28 | ports:
29 | | - protocol: TCP
30 | | | port: 80
31 | | | targetPort: 80
32 | | | nodePort: 30081
33 | type: NodePort
```

Time series churn

- When old series are substituted by new ones
- Monitoring solutions don't like high churn rate :(



Time series churn

- K8s may generate high churn rate for active time series because of:
 - Frequent deployments
 - Frequent pod auto-scale events



Time series churn

- Every new deployment or just changing an image tag results into a new set of time series
- The number of new time series per each deployment or HPA event can be estimated as:

`deployment * replicas * (container stat metrics + application metrics)`

Do we need all these metrics?

Do we need all these metrics?

- No easy answer :(



Do we need all these metrics?

- No easy answer :(
- “No” - our monitoring system uses only a small fraction of the collected metrics in alerting/recording rules and dashboards

Do we need all these metrics?

- No easy answer :(
- “No” - our monitoring system uses only a small fraction of the collected metrics in alerting/recording rules and dashboards
- “Yes” - the need for unused metrics may arise in the future

How to determine
the exact set of needed metrics?



How to determine the exact set of needed metrics?

Mimirtool from Grafana?

In an *unloaded* 3-node Kubernetes cluster, Kube-Prometheus will ship roughly **40k** active series by default. The following allowlist configuration should reduce this volume to roughly **8k** active series.

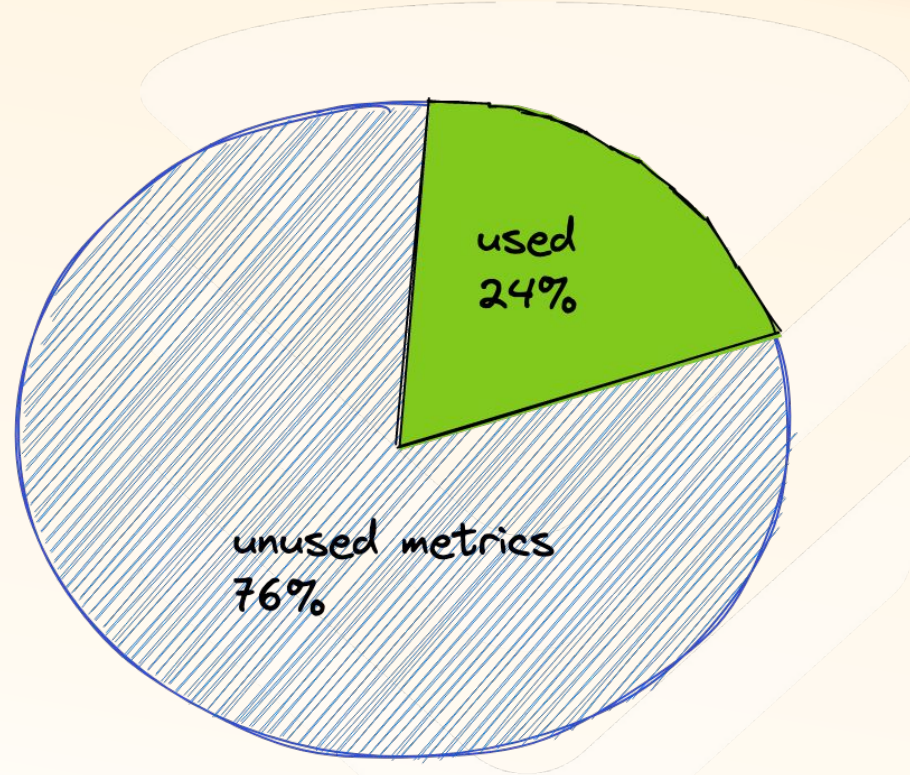
Source: https://grafana.com/docs/grafana-cloud/kubernetes/prometheus/helm-operator-migration/reduce_usage/

Millions of metrics

Existing solutions like kube-prometheus-stack collect too many metrics and most of them are collected just in case without utility in practice

1277 unique metric names exposed
by 1 k8s node in prometheus-stack

307 of these metrics are actually
used in rules and Grafana dashboards



Monitoring standards?

- There is no an established standard for metrics (in K8s too)

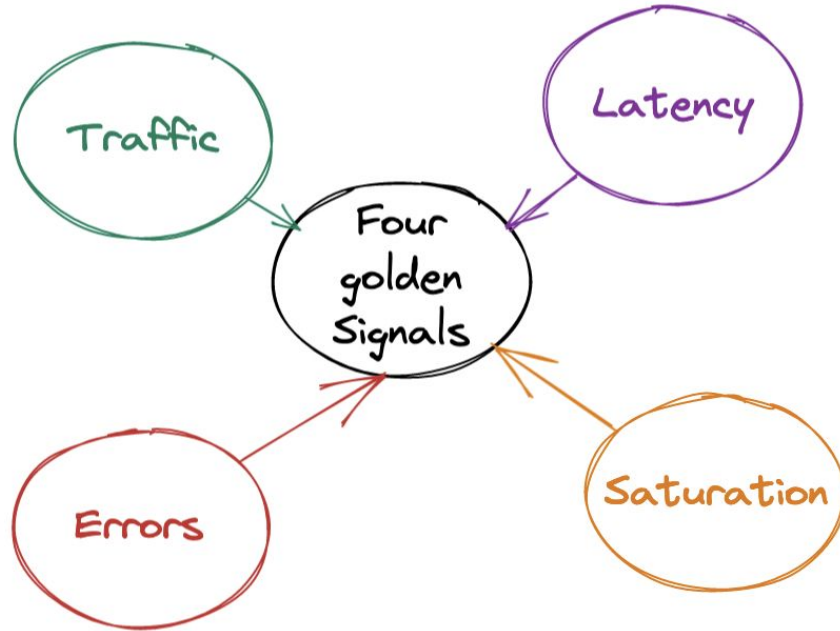


Monitoring standards?

- There is no an established standard for metrics (in K8s too)
- Community and many companies try to invent own standards

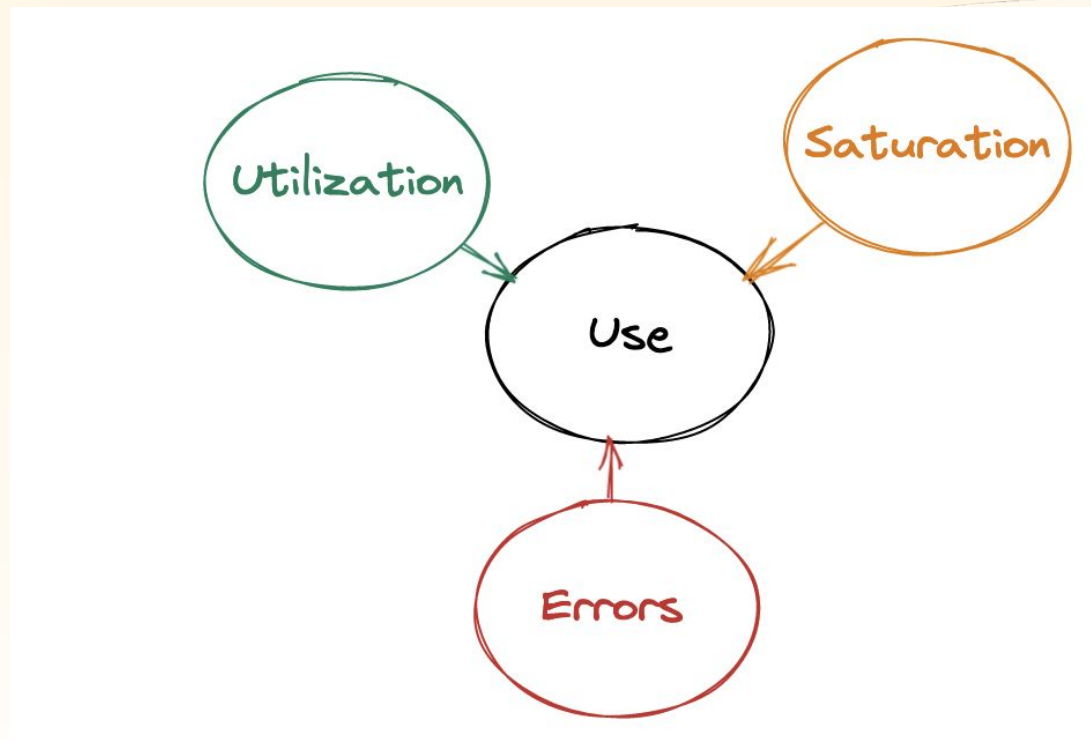
Monitoring standards?

The Four Golden Signals method by Google



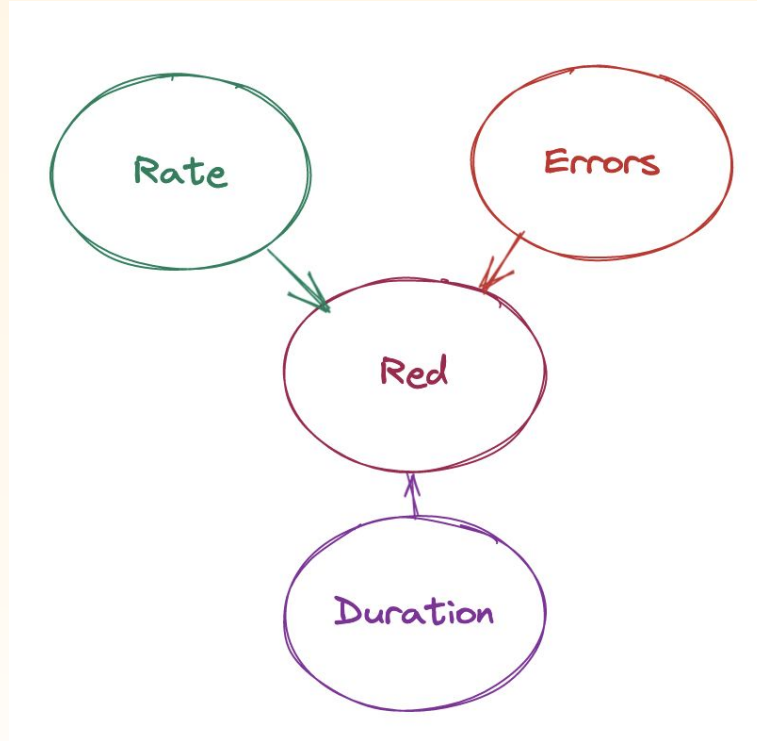
Monitoring standards?

USE method by Brendan Gregg



Monitoring standards?

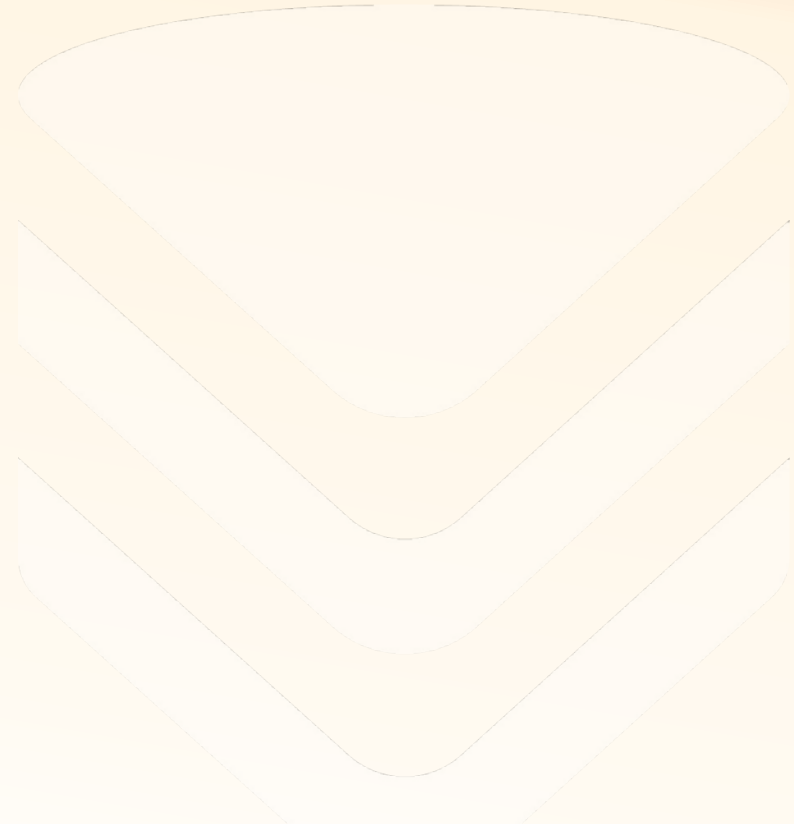
RED method by Weave.works



Monitoring standards?

This situation leads to

- so many of metrics in various applications



Monitoring standards?

This situation leads to

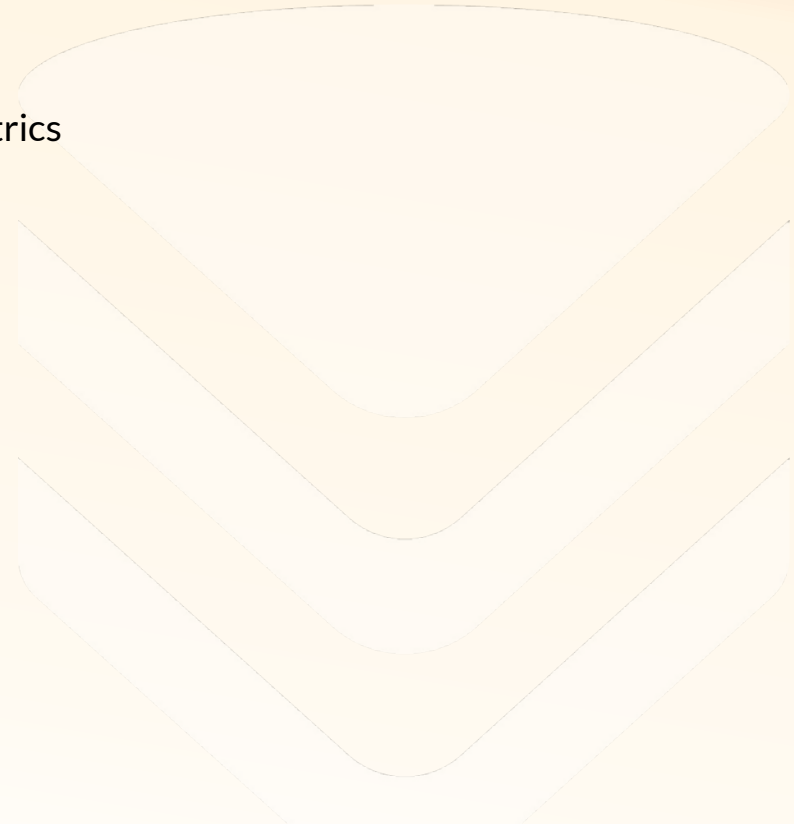
- so many of metrics in various applications
- metrics change over time



Monitoring standards?

This situation leads to

- so many of metrics in various applications
- metrics change over time
- many articles and opinions about most essential metrics



Monitoring standards?

Community dashboards for kubernetes at grafana.com are mostly outdated

The image shows two screenshots of the Grafana.com dashboard marketplace. The top screenshot displays a dashboard titled "1. Kubernetes Deployment Statefulset Daemonset metrics" by user pratt0318. It has 867157 downloads and 12 reviews, with a 4-star rating. The description states it monitors Kubernetes deployments using Prometheus and cAdvisor metrics. The bottom screenshot shows a dashboard titled "Cluster Monitoring for Kubernetes" by Pivotal Observability. It has 539509 downloads and 9 reviews, with a 4.5-star rating. The description mentions it provides cluster health info and can be deployed with PSPs. Both dashboards are listed as "DASHBOARD" type and include instructions on how to start with Grafana Cloud.

1. Kubernetes Deployment Statefulset Daemonset metrics by pratt0318
DASHBOARD
Monitors Kubernetes deployments in cluster using Prometheus. Shows overall cluster CPU / Memory of deployments, replicas in each deployment. Uses Kube state metrics and cAdvisor metrics (741)
Last updated: 4 years ago
Start with Grafana Cloud and the new FREE tier. Includes 10K series Prometheus or Graphite Metrics and 50gb Loki Logs

Downloads: 867157
Reviews: 12
★★★★☆
Add your review!

Overview Revisions Reviews

Dashboard Revisions

Revision	Description
1	Monitors Kubernetes deployments in cluster using Prometheus. Shows overall cluster CPU / Memory of deployment. Uses Kube state metrics and cAdvisor metrics

Cluster Monitoring for Kubernetes by Pivotal Observability
DASHBOARD
This dashboard provides cluster admins with the ability to monitor nodes and identify workload bottlenecks. It can be deployed with PSPs enabled using the following helm chart - <https://github.com/pivotal-cf/charts-grafana>
Last updated: 3 years ago
Start with Grafana Cloud and the new FREE tier. Includes 10K series Prometheus or Graphite Metrics and 50gb Loki Logs

Get this dashboard:
Downloads: 539509
Reviews: 9
★★★★☆
Add your review!

Overview Revisions Reviews

Dashboard Revisions

Revision	Description	Created	Download
1	Cluster health info	April 2nd 2019, 11:14 pm	10000

Observability challenges in microservices architecture

- Every microservice instance needs own metrics



Observability challenges in microservices architecture

- Every microservice instance needs own metrics
- Users need to track and correlate events across multiple services



Observability challenges in microservices architecture

- Every microservice instance needs own metrics
- Users need to track and correlate events across multiple services
- Ephemerality of the services only makes situation worse

Observability challenges in microservices architecture

- Every microservice instance needs own metrics
- Users need to track and correlate events across multiple services
- Ephemerality of the services only makes situation worse
- New entities like distributed traces are needed to improve the situation

Observability challenges in microservices architecture

- Every microservice instance needs own metrics
- Users need to track and correlate events across multiple services
- Ephemerality of the services only makes situation worse
- New entities like distributed traces are needed to improve the situation
- Network issues come into the play

Observability challenges in microservices architecture

- Every microservice instance needs own metrics
- Users need to track and correlate events across multiple services
- Ephemerality of the services only makes situation worse
- New entities like distributed traces are needed to improve the situation
- Network issues come into the play
- Service collocation on one node create a "noisy neighbour" problem

Observability challenges in microservices architecture

- Every microservice instance needs own metrics
- Users need to track and correlate events across multiple services
- Ephemerality of the services only makes situation worse
- New entities like distributed traces are needed to improve the situation
- Network issues come into the play
- Service collocation on one node create a "noisy neighbour" problem
- Service mesh introduces yet another layer, which needs to be monitored

How k8s affects the monitoring

- Kubernetes has a dark side of increasing complexity and metrics footprint



How k8s affects the monitoring

- Kubernetes has a dark side of increasing complexity and metrics footprint
- Current monitoring solutions are busy with overcoming complexities introduced by k8s:
 - Active time series churn (Ephemerality)
 - Huge volumes of metrics for each layer and service

How k8s affects the monitoring

- Kubernetes has a dark side of increasing complexity and metrics footprint
- Current monitoring solutions are busy with overcoming complexities introduced by k8s:
 - Active time series churn (Ephemerality)
 - Huge volumes of metrics for each layer and service
- Thousands of hours were spent just to adapt existing monitoring tools for k8s

How k8s affects the monitoring

- Kubernetes has a dark side of increasing complexity and metrics footprint
- Current monitoring solutions are busy with overcoming complexities introduced by k8s:
 - Active time series churn (Ephemerality)
 - Huge volumes of metrics for each layer and service
- Thousands of hours were spent just to adapt existing monitoring tools for k8s
- Maybe, if there was no k8s, we won't need distributed traces and exemplars?

How k8s affects the monitoring

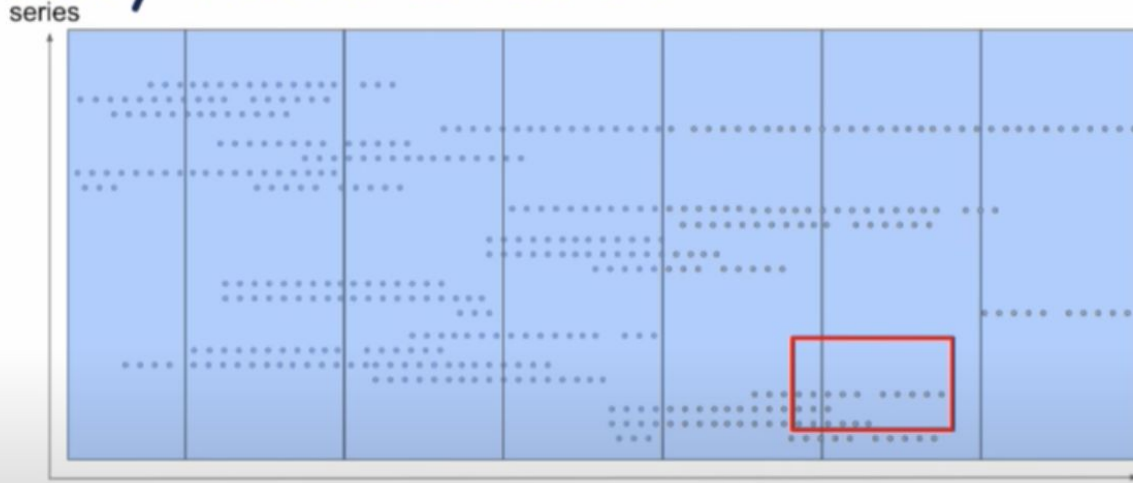
- Kubernetes has a dark side of increasing complexity and metrics footprint
- Current monitoring solutions are busy with overcoming complexities introduced by k8s:
 - Active time series churn (Ephemerality)
 - Huge volumes of metrics for each layer and service
- Thousands of hours were spent just to adapt existing monitoring tools for k8s
- Maybe, if there was no k8s, we won't need distributed traces and exemplars?
- Maybe, if there was no k8s, all that time spent on overcoming those difficulties could be invested in more useful observability tools such as automated root cause analysis and metrics' correlation?

How Kubernetes deals with millions of metrics?

- Some metrics can be disabled via command line flag `--disabled-metrics`
- The list of allowed label values can be specified via `--allow-label-value`

How Prometheus deals with k8s challenges?

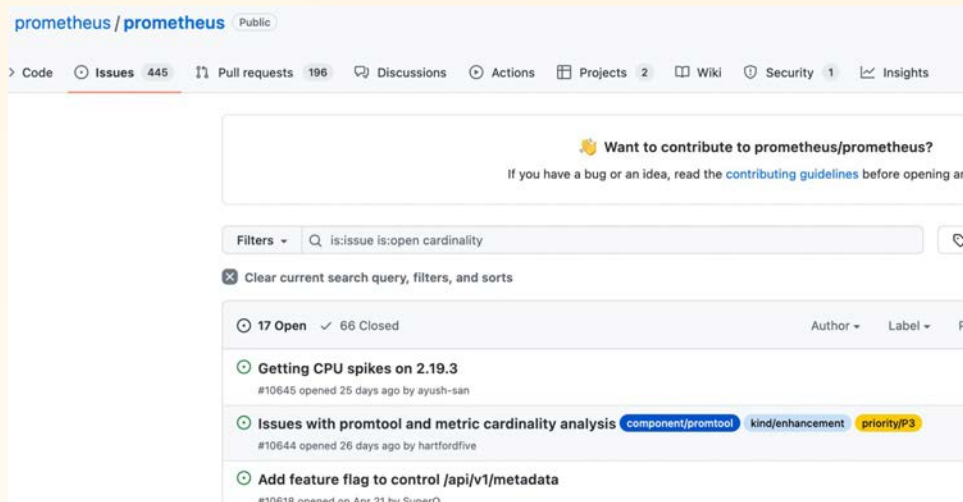
Why Time-shard?



<https://prometheus.io/blog/2017/11/08/announcing-prometheus-2-0/>

How Prometheus deals with k8s challenges?

But still, there a lot of issues with churn rate and cardinality



prometheus / prometheus Public

> Code Issues 445 Pull requests 196 Discussions Actions Projects 2 Wiki Security 1 Insights

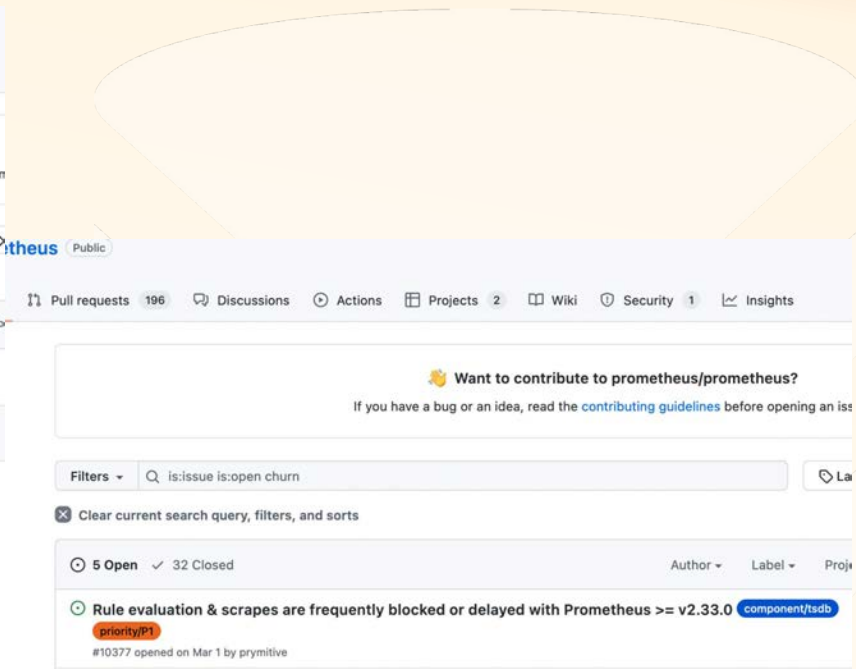
Want to contribute to prometheus/prometheus?
If you have a bug or an idea, read the [contributing guidelines](#) before opening an

Filters

Clear current search query, filters, and sorts

17 Open ✓ 66 Closed Author Label P

- Getting CPU spikes on 2.19.3
#10645 opened 25 days ago by ayush-san
- Issues with promtool and metric cardinality analysis component/promtool kind/enhancement priority/P3
#10644 opened 26 days ago by hartfordfive
- Add feature flag to control /api/v1/metadata
#10618 opened on Apr 21 by SuserO



prometheus / prometheus Public

Pull requests 196 Discussions Actions Projects 2 Wiki Security 1 Insights

Want to contribute to prometheus/prometheus?
If you have a bug or an idea, read the [contributing guidelines](#) before opening an iss

Filters

Clear current search query, filters, and sorts

5 Open ✓ 32 Closed Author Label Proj

- Rule evaluation & scrapes are frequently blocked or delayed with Prometheus >= v2.33.0 component/tsdb priority/P1
#10377 opened on Mar 1 by prymitive

How VictoriaMetrics deals with k8s challenges?

- VictoriaMetrics was born to address high cardinality issues of Prometheus v1



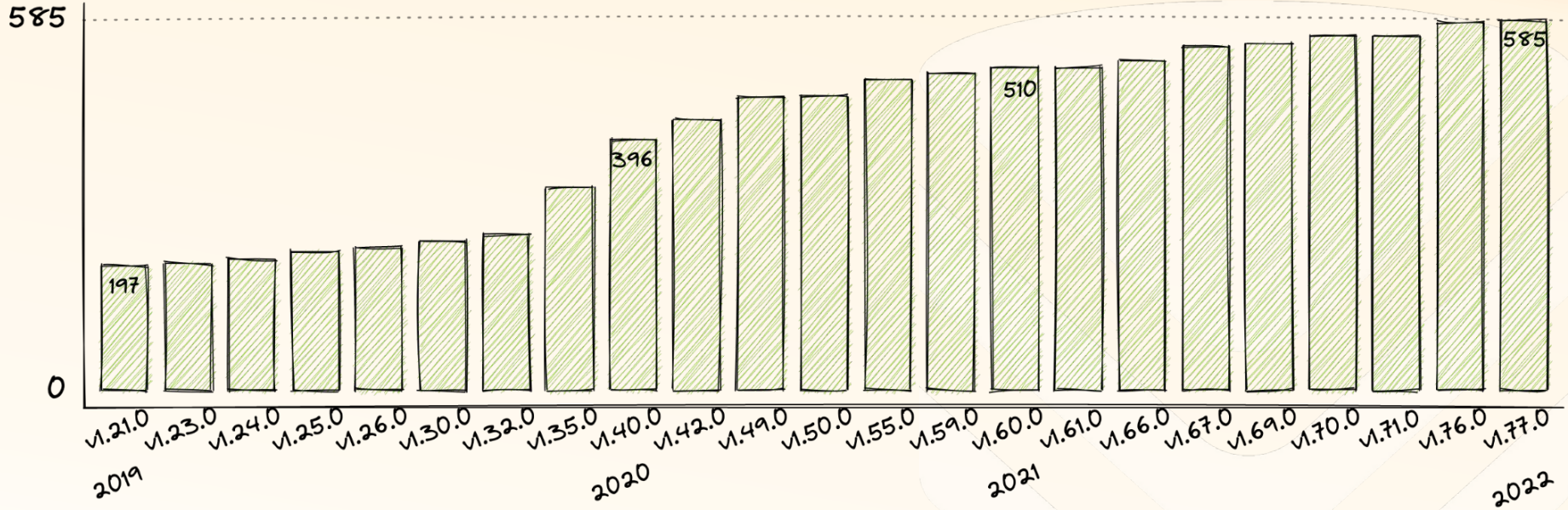
How VictoriaMetrics deals with k8s challenges?

- VictoriaMetrics was born to address high cardinality issues of Prometheus v1
- It is optimized for using lower RAM and disk space for high cardinality series

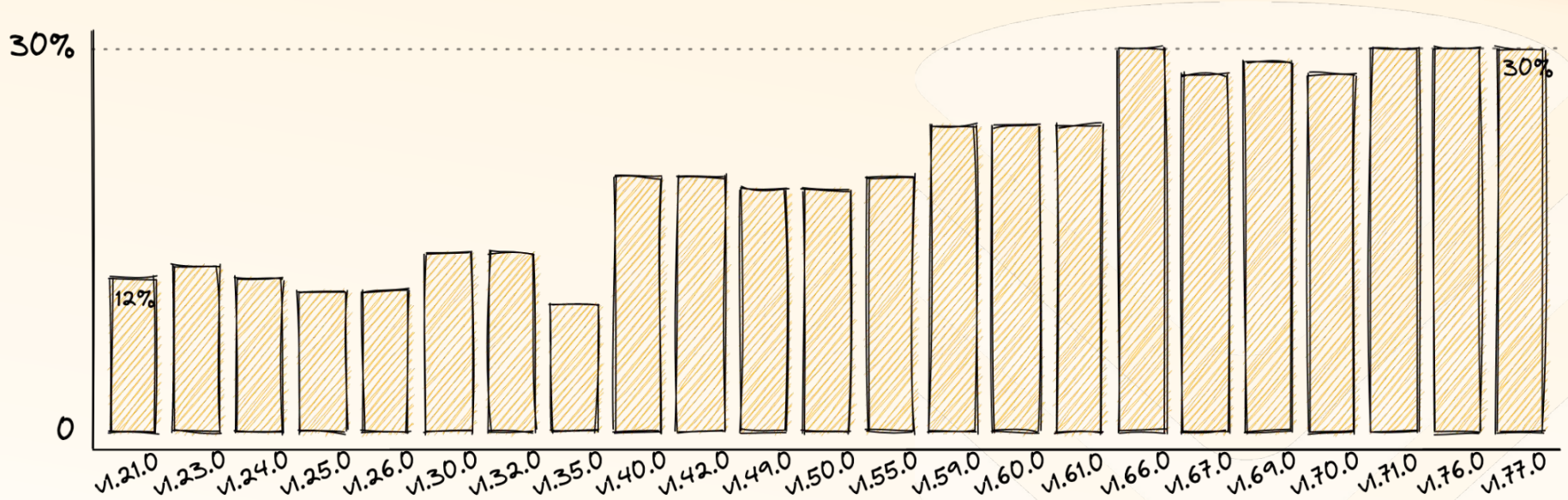
How VictoriaMetrics deals with k8s challenges?

- VictoriaMetrics was born to address high cardinality issues of Prometheus v1
- It is optimized for using lower RAM and disk space for high cardinality series
- It uses per-day inverted index in order to overcome time series churn

Metric names for VictoriaMetrics - 3x growth



The percentage of used metrics in VictoriaMetrics



How can we improve the situation

- K8s monitoring complexity must be reduced



How can we improve the situation

- K8s monitoring complexity must be reduced
- The number of exposed metrics must be reduced



How can we improve the situation

- K8s monitoring complexity must be reduced
- The number of exposed metrics must be reduced
- The number of histograms must be reduced



How can we improve the situation

- Histograms generate huge amounts of series

```
# HELP prometheus_http_request_duration_seconds Histogram of latencies for
HTTP requests.
# TYPE prometheus_http_request_duration_seconds histogram
prometheus_http_request_duration_seconds_bucket(handler="/",le="0.1") 25547
prometheus_http_request_duration_seconds_bucket(handler="/",le="0.2") 26688
prometheus_http_request_duration_seconds_bucket(handler="/",le="0.4") 27760
prometheus_http_request_duration_seconds_bucket(handler="/",le="1") 28641
prometheus_http_request_duration_seconds_bucket(handler="/",le="3") 28782
prometheus_http_request_duration_seconds_bucket(handler="/",le="8") 28844
prometheus_http_request_duration_seconds_bucket(handler="/",le="20") 28855
prometheus_http_request_duration_seconds_bucket(handler="/",le="60") 28860
prometheus_http_request_duration_seconds_bucket(handler="/",le="120") 28860
prometheus_http_request_duration_seconds_bucket(handler="/",le="+Inf")
28860
prometheus_http_request_duration_seconds_sum(handler="/") 1863.80491025699
prometheus_http_request_duration_seconds_count(handler="/") 28860
```

How can we improve the situation

- K8s monitoring complexity must be reduced
- The number of exposed metrics must be reduced
- The number of histograms must be reduced
- The number of per-metric labels must be reduced (pod-level labels?)

How can we improve the situation

- K8s monitoring complexity must be reduced
- The number of exposed metrics must be reduced
- The number of histograms must be reduced
- The number of per-metric labels must be reduced (pod-level labels?)
- Time series churn rate must be reduced (HPA, deployments?)

How can we improve the situation

- K8s monitoring complexity must be reduced
- The number of exposed metrics must be reduced
- The number of histograms must be reduced
- The number of per-metric labels must be reduced (pod-level labels?)
- Time series churn rate must be reduced (HPA, deployments?)
- The community will come up with a standard for k8s monitoring - let's do it together!

Questions?

<https://victoriametrics.com/blog>

<https://github.com/VictoriaMetrics>

<https://github.com/valyala>

