

Open Source Cloud-Native Observability with VictoriaMetrics and OpenTelemetry

Cloud Native Valencia | 27 November 2025
Diana Todea - DX Engineer

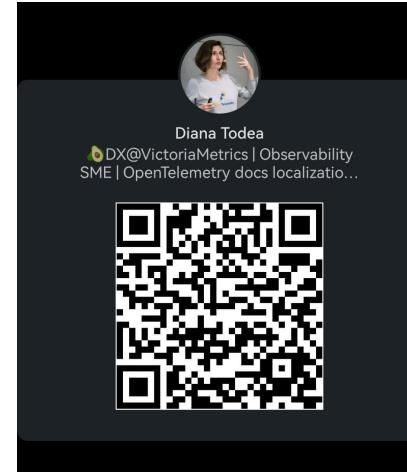




OpenTelemetry member and contributor

Cloud Native Days Romania organizer

Co-lead CNCF Merge-Forward Neurodiversity



https://github.com/didiViking/Conferences_Talks

What VictoriaMetrics isn't

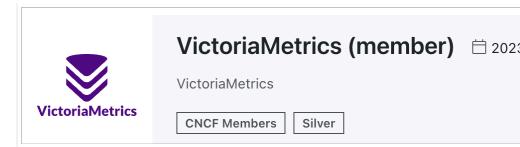
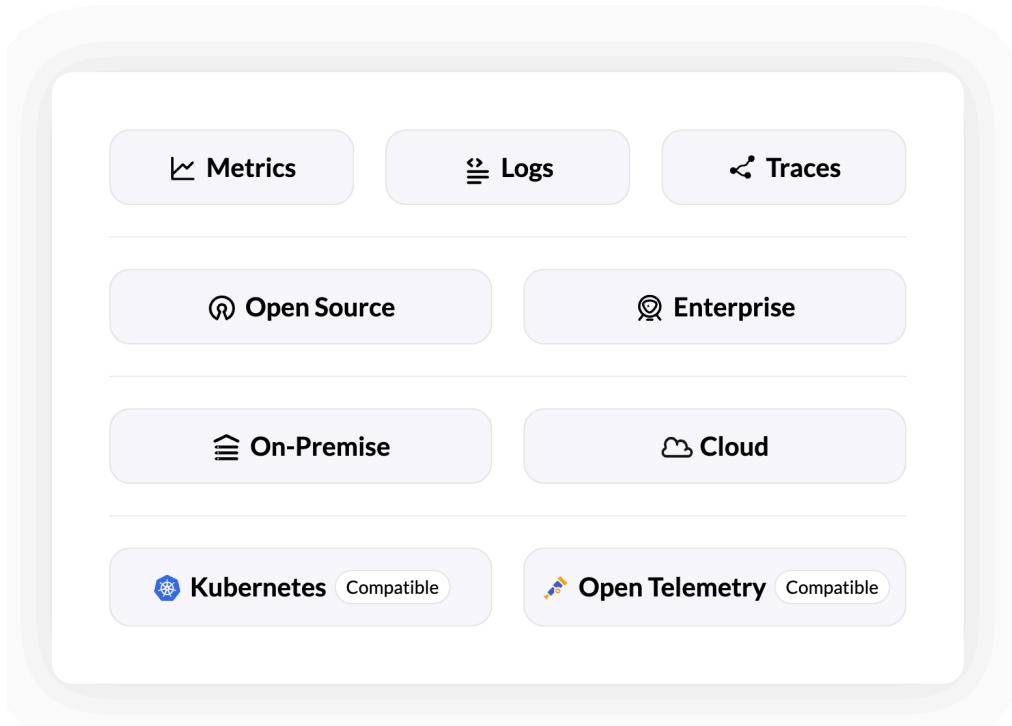
A fork of Prometheus CNCF project

Named after Queen Victoria or Victoria British Columbia

Just a frontend visualization tool running ClickHouse or Thanos in the backend



VictoriaMetrics Observability Stack



VictoriaMetrics and its Open-Source strategy

How the co-founders built and tried to sell VictoriaMetrics in the beginning

They decided to create an Open Source community and build excellent products

No investors led strategy, no obligation to close the OSS project



https://youtu.be/-DbbIZzFHIY?si=BKJ9njiTNhdY_xyv

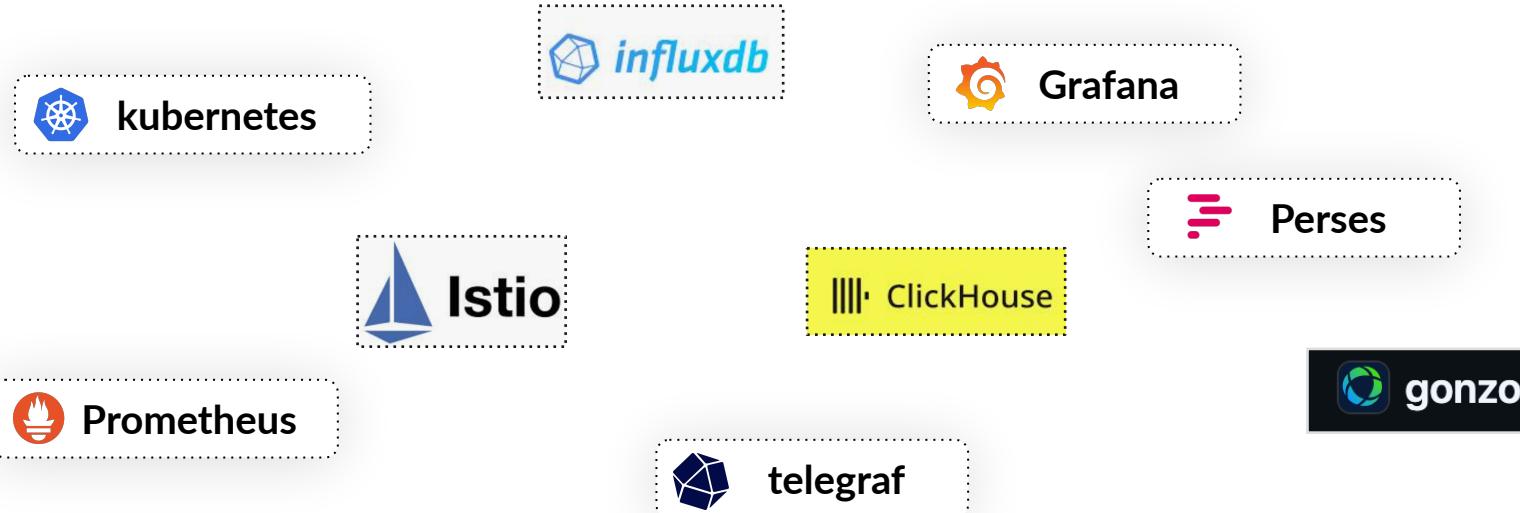


VictoriaMetrics

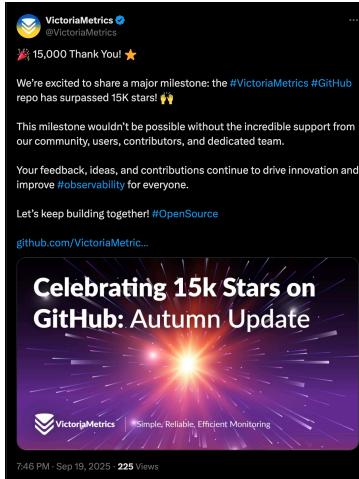
Simple, Reliable, Efficient Monitoring

victoriametrics.com

VictoriaMetrics contributes to OSS and Cloud Native projects



VictoriaMetrics hits 1 billion downloads and 15k+ stars on GitHub



VictoriaMetrics Public

Watch 150 Fork 1.5k Starred 10.4k

About

VictoriaMetrics: fast, cost-effective monitoring solution and time series database

victoriametrics.com/

Codebase

Issues

Pulls

Commits

Releases

Activity

Custom properties

15.4k stars

15.6k watching

1.5k forks

Report repository

Contributing

Security policy

Code of conduct

Apache-2.0 license

169 Branches 924 Tags

Go to file Add file Add Code

master adGabb 3 days ago 11,632 Commits

rtm0 lib/storage: minor metricNameSearch fixes (#10065) 3 weeks ago

.github make: include s390x binaries into release artifacts (#9941) 3 weeks ago

app app/vmalert: do not increment errors counter on cancel c... 5 days ago

apptest apptest: add metrics metadata test for vmsingle last week

benchmarks benchmark: add gnuplot to show write speed (#9490) 2 months ago

codespell spellcheck: run 5 months ago

dashboards dashboards: run rake dashboards-sync last month

deployment deployment/docker: update VM components version to v1... last week

docs docs: add warningblockquote regarding latest backup lf... 4 days ago

lib lib/storage: minor metricNameSearch fixes (#10065) 3 days ago

package simplify release process (#3012) 3 years ago

ports/OpenBSD docs: convert png images to webp in all the docs except ... 2 years ago

vendor vendor: update github.com/valyala/gozstd from v1.23.2 to... 2 weeks ago

.dockernignore added packer build for DigitalOcean Droplets (#1917) 4 years ago

.gitignore app: add vlagent component 5 months ago

.golangci.yml ci: golangci-lint 1.6.x -> 2.2.1 4 months ago

.whited.yml add MPL-2.0 to approved licenses last year

New kids on the block: VictoriaLogs and VictoriaTraces

The image shows two GitHub repository pages side-by-side:

- VictoriaLogs (Public)**:
 - Repository Overview:** master branch, 13 branches, 19 tags, 11,581 commits.
 - Issues:** 837 open issues.
 - Code:** Go file, Add file, Code view.
 - Topics:** kubernetes, elasticsearch, grafana, logs, siem, loki, observability, opentelemetry.
 - Files:** .github, app, apptest, benchmarks/gharchive, codespell, dashboards, deployment, docs, lib, package/release, vendor, .dockercfgignore, .gitignore, .golangci.yml, .wwhred.yml, CODE_OF_CONDUCT.md.
 - Commits:** 6e49f64 - 3 days ago, 11,581 Commits.
 - Contributors:** 48 forks.
 - Releases:** v1.38.0 (Latest), last week, + 18 releases.
- VictoriaTraces (Public)**:
 - Repository Overview:** master branch, 17 branches, 7 tags, 11,171 commits.
 - Issues:** 62 open issues.
 - Code:** Go file, Add file, Code view.
 - Topics:** Apache-2.0 license, Code of conduct, Contributing, Security policy, Activity, Custom properties.
 - Files:** .github, app, apptest, codespell, dashboards, deployment/docker, docs, lib, package/release, vendor, .dockercfgignore, .gitignore, .golangci.yml, .wwhred.yml, CODE_OF_CONDUCT.md, CONTRIBUTING.md, LICENSE, Makefile, README.md.
 - Commits:** 100ea97 - 1 hour ago, 11,171 Commits.
 - Contributors:** 16 forks.
 - Releases:** v0.5.1 (Latest), 5 days ago, + 6 releases.
 - Packages:** No packages published. Publish your first package.
 - Contributors:** 16.

VictoriaMetrics Sustainable Features

1.7x

less memory

90%

energy cost
reduction

2.5x

less disk space

10x

infrastructure
cost Savings

16x

faster query
latency

4x

network cost
Savings

CNCF Green Reviews WG

The CNCF Green Reviews Working Group (WG) is an open-source, community-led project that is part of the [CNCF Environmental Sustainability Technical Advisory Group \(TAG ENV\)](#).

The aim of the Green Reviews WG is to set up infrastructure to measure the sustainability footprint of [CNCF Projects](#).

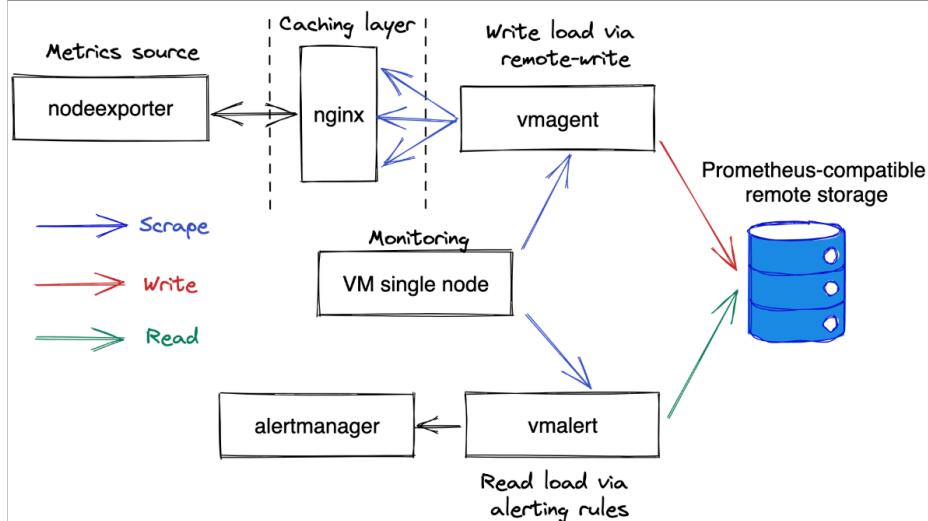
Measuring the sustainability footprint of software is not an easy task. Our vision is that the WG will compute the sustainability data for every release of a CNCF project that requests a sustainability footprint assessment. To achieve such a vision, our goal is to develop a workflow that can integrate well with the existing software lifecycle of other CNCF projects.

A good way to practically understand the first version of the workflow that the WG is designing is to take a look at this simplified architecture diagram:

★ <https://github.com/cncf-tags/green-reviews-tooling>



Run your own benchmarks!



★ [https://github.com/
VictoriaMetrics/prometheus-
benchmark](https://github.com/VictoriaMetrics/prometheus-benchmark)

★ [https://victoriametrics.com/
blog/benchmark-100m/](https://victoriametrics.com/blog/benchmark-100m/)

VictoriaLogs Sustainable Features

1000x

boosts haystack
search speed

30x

less RAM

15x

less disk
space



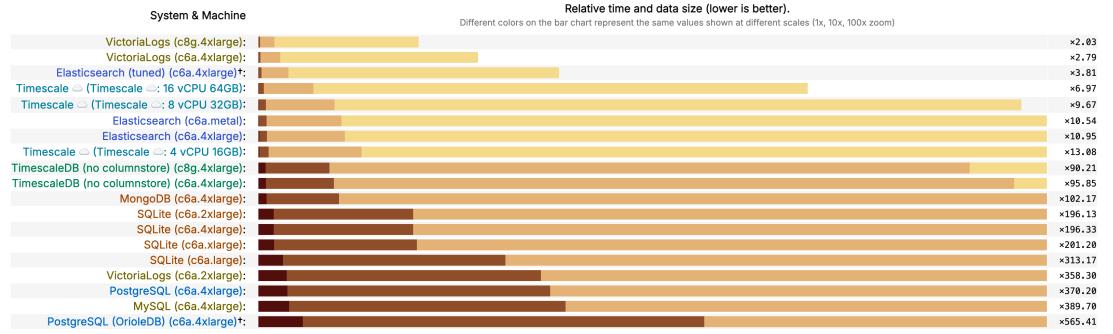
Benchmarks

Improved Query Performance

Resource Efficiency



VictoriaLogs vs. Elasticsearch vs. MongoDB etc.



Detailed Comparison

	VictoriaLogs (c8g.4xlarge)	VictoriaLogs (c6a.4xlarge)	Elasticsearch (tuned) (c6a.4xlarge)	Timescale (Timescale : 16 vCPU 64GB)	Timescale (Timescale : 8 vCPU 32GB)	Elasticsearch (c6a.metal)	Elasticsearch (c6a.4xlarge)	Timescale (Timescale : 4 vCPU 16GB)	TimescaleDB (c8g.4xlarge)
Load time:	1055s (+1, 13)	1058s (+1, 12)	7140s (+7, 62)	3582s (+3, 82)	4180s (+4, 38)	9565s (+10, 21)	9531s (+10, 17)	4536s (+4, 84)	1055s (+1, 13)
Data size:	15.88 GB (+1, 08)	15.81 GB (+1, 08)	106.47 GB (+6, 08)	17.98 GB (+1, 14)	78.85 GB (+5, 08)	79.96 GB (+5, 07)	17.98 GB (+1, 14)	17.98 GB (+1, 14)	15.88 GB (+1, 08)
Q0.	0.045s (+2, 39)	0.068s (+3, 10)	0.817s (+1, 20)	0.064s (+3, 28)	0.063s (+3, 27)	0.032s (+1, 00)	0.024s (+1, 49)	0.033s (+4, 11)	0.045s (+2, 39)
Q1.	0.026s (+2, 28)	0.037s (+2, 83)	0.087s (+1, 05)	0.252s (+15, 86)	0.373s (+23, 21)	0.886s (+1, 00)	0.044s (+3, 24)	3.431s (+208, 55)	0.026s (+2, 28)
Q2.	0.923s (+5, 07)	0.574s (+3, 17)	0.896s (+4, 93)	0.322s (+1, 88)	0.594s (+2, 79)	0.575s (+3, 18)	0.630s (+3, 59)	3.901s (+21, 26)	0.923s (+5, 07)
Q3.	1.018s (+5, 91)	0.798s (+4, 69)	1.558s (+9, 01)	0.261s (+1, 56)	0.411s (+2, 42)	0.249s (+1, 44)	0.689s (+3, 97)	4.422s (+25, 47)	1.018s (+5, 91)
Q4.	1.113s (+3, 77)	2.121s (+7, 16)	8.583s (+1, 72)	14.785s (+9, 42)	22.495s (+75, 59)	8.290s (+1, 81)	8.297s (+1, 83)	22.636s (+76, 86)	1.113s (+3, 77)
Q5.	0.929s (+1, 74)	1.637s (+3, 06)	1.458s (+2, 72)	21.977s (+40, 81)	43.578s (+80, 91)	8.529s (+1, 80)	0.745s (+1, 49)	58.296s (+108, 22)	0.929s (+1, 74)
Q6.	0.048s (+3, 72)	0.657s (+5, 88)	0.439s (+33, 28)	8.228s (+17, 08)	0.318s (+24, 33)	0.252s (+19, 43)	0.386s (+23, 39)	2.393s (+178, 84)	0.048s (+3, 72)
Q7.	0.025s (+1, 88)	0.035s (+1, 29)	0.265s (+7, 98)	0.238s (+7, 14)	0.325s (+9, 03)	0.903s (+26, 28)	0.887s (+25, 88)	0.468s (+13, 76)	0.025s (+1, 88)
Q8.	3.364s (+1, 08)	2.168s (+1, 59)	2.391s (+1, 75)	25.255s (+18, 39)	80.955s (+89, 94)	9.155s (+6, 67)	8.898s (+6, 48)	91.126s (+66, 34)	3.364s (+1, 08)
Q9.	1.888s (+1, 08)	2.981s (+1, 68)	3.333s (+1, 84)	29.373s (+16, 14)	88.163s (+48, 58)	13.552s (+7, 46)	12.829s (+7, 06)	97.452s (+53, 61)	1.888s (+1, 08)
Q10.	0.279s (+1, 07)	0.434s (+1, 64)	0.3085s (+1, 10)	2.592s (+9, 28)	9.625s (+35, 59)	1.154s (+4, 30)	1.258s (+4, 69)	13.087s (+48, 37)	0.279s (+1, 07)
Q11.	8.336s (+1, 06)	8.501s (+1, 56)	8.359s (+1, 13)	3.282s (+10, 05)	13.538s (+41, 37)	1.410s (+4, 34)	1.592s (+4, 89)	16.479s (+58, 35)	8.336s (+1, 06)
Q12.	0.769s (+2, 39)	1.290s (+3, 99)	8.318s (+1, 00)	6.921s (+21, 28)	7.551s (+23, 21)	4.271s (+13, 14)	4.254s (+13, 09)	15.055s (+46, 25)	0.769s (+2, 39)

★ https://benchmark.clickhouse.com/#system=+lrc|ehed|noB|yLgS|lgQ|Lt|m%E2%98%81%20nu|coog&type=-&machine=-&cluster_size=-&opensource=-&tuned=-&metric=combined&queries=-

★ <https://github.com/ClickHouse/ClickBench/>

VictoriaLogs vs. DuckDB vs. PostgreSQL

JSONBench — a Benchmark For Data Analytics On JSON



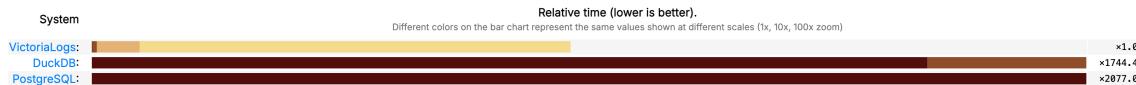
[Methodology](#) | [Reproduce and Validate the Results](#) | [Add a System](#) | [Report Mistake](#) | See also: [ClickBench](#)

System: All ClickHouse Apache Doris DuckDB Elasticsearch (no source) Elasticsearch GreptimeDB MongoDB PostgreSQL SingleStore Starrocks VictoriaLogs

Scale: 1 million 10 million 100 million 1000 million

Metric: Cold Run Hot Run Storage Size Data Quality

Flattening: No Yes



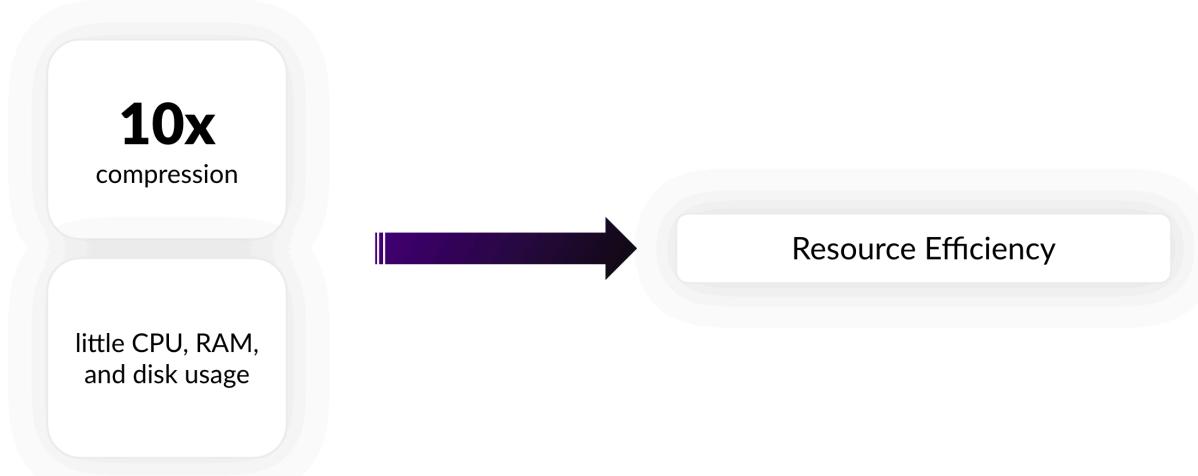
Detailed Comparison

	VictoriaLogs	DuckDB	PostgreSQL
Data size:	113.49 GiB (+1.00)	448.14 GiB (+3.88)	615.91 GiB (+5.42)
Data quality:	999999243	974400000	804000000
Q1.	0.028s (+1.00)	3717.611s (+x97832.13)	3884.170s (+x102215.26)
Q2.	19.794s (+1.00)	3721.045s (+x187.89)	4277.808s (+x216.01)
Q3.	1.687s (+1.00)	3717.631s (+x299.19)	4253.340s (+x2638.49)
Q4.	5.185s (+1.00)	3719.273s (+x715.94)	4907.090s (+x944.58)
Q5.	6.963s (+1.00)	3722.804s (+x333.89)	4913.508s (+x04.65)

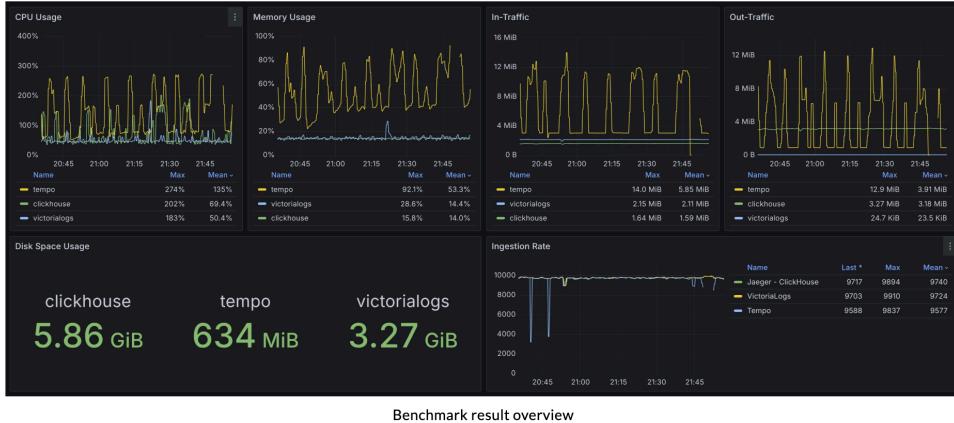
★ <https://github.com/ClickHouse/JSONBench>

★ <https://docs.victoriametrics.com/victorialogs/#benchmarks>

VictoriaTraces Sustainable Features



VictoriaLogs vs. Clickhouse vs. Grafana Tempo



ⓘ Why is there no benchmark result for Jaeger + Elasticsearch? And why use 10,000 spans/s?

The Elasticsearch-based solution began to crash at an ingestion rate as around 5,000 spans/s, and it couldn't remain stable under higher loads with default settings. That's why it's excluded from the final comparison.

We conducted multiple benchmarks under different loads to ensure all competitors could run smoothly. If you're interested in these more extreme benchmark cases, feel free to comment below or join our [Slack channel](#) for discussion.

★ 3.7x less RAM and up to 2.6x less CPU

★ <https://victoriametrics.com/blog/dev-note-distributed-tracing-with-victorialogs/>

Time Series Database - Ranking

Last 28 days / Monthly ranking of repos in this collection by stars, pull requests, issues. Historical Ranking by Popularity.

Last 28 Days / Month-to-Month Ranking

The following table ranks repositories using three metrics: stars, pull requests, and issues. The table compares last 28 days or the most recent two months of data and indicates whether repositories are moving up or down the rankings.

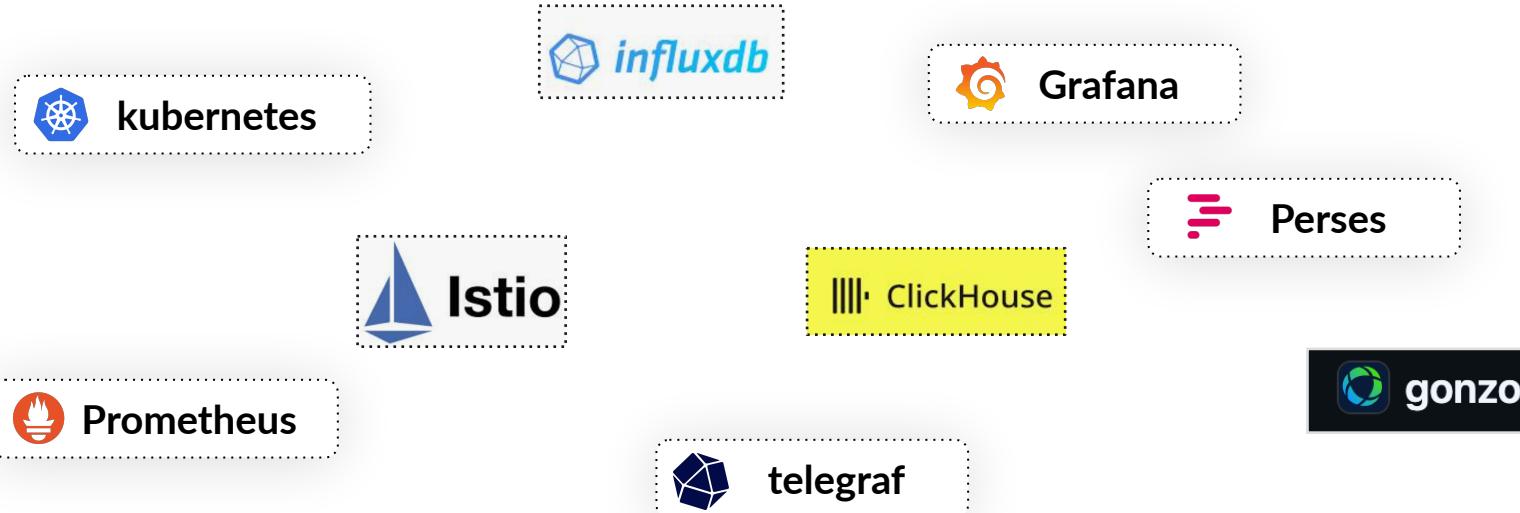
★ Stars毖 Pull Requests⌚ IssuesLast 28 DaysMonth-to-Month

SHOW SQL <>

Last 28 days Ranking - Stars				
Last 28 Days	Repository	Stars	Total	
1	ClickHouse/ClickHouse	232 ↑25.4%	43.31k	
2	prometheus/prometheus	208 ↑34.2%	62.1k	
3	netdata/netdata	145 ↑15.1%	80.23k	
4	timescale/timescaledb	125 ↑20.2%	19.98k	
5 ↑2	VictoriaMetrics/VictoriaMetrics	76 ↑38.2%	14.58k	
6 ↓1	surrealdb/surrealdb	74 ↑23.3%	30.04k	
7 ↑4	influxdata/influxdb	49 ↑104.2%	32.49k	
8 ↑5	GreptimeTeam/greptimedb	43 ↑126.3%	5.24k	
9 ↓3	questdb/questdb	39 ↓31.6%	16.33k	
10 ↓2	citusdata/citus	33 ↓15.4%	11.67k	

★ <https://ossinsight.io/collections/time-series-database>

VictoriaMetrics contributes to OSS and Cloud Native projects



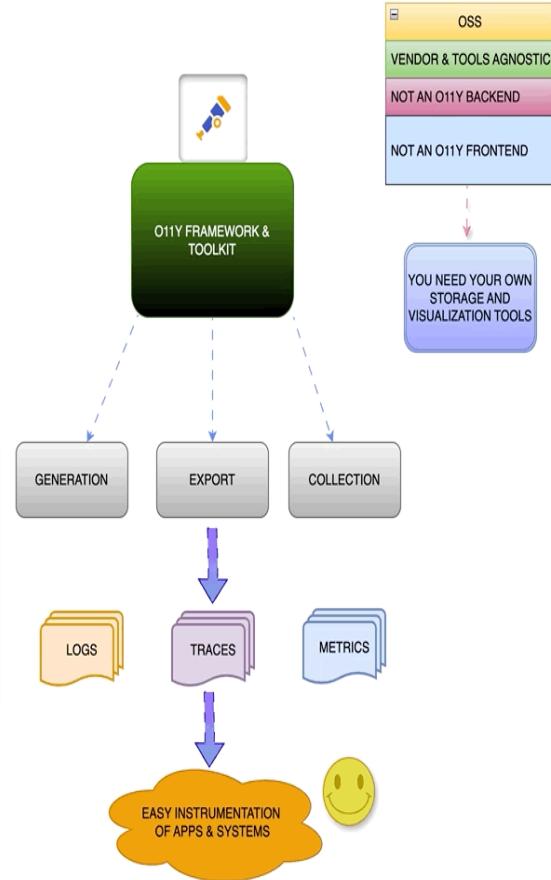
What is OpenTelemetry



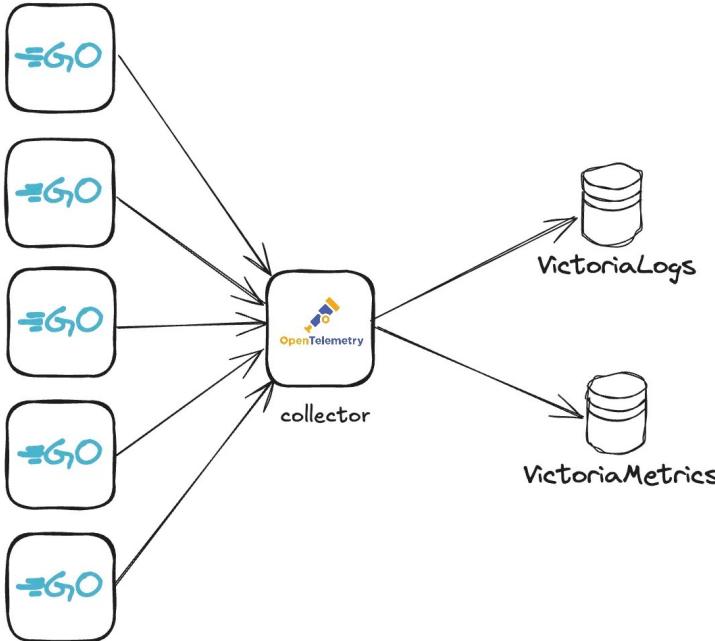
OpenTelemetry community content

OpenTelemetry was accepted to CNCF on May 7, 2019 and moved to the **Incubating** maturity level on August 26, 2021.

[VISIT PROJECT WEBSITE](#)



VictoriaMetrics + OpenTelemetry



VictoriaMetrics and VictoriaLogs support ingestion of metrics and logs in the OTEL format.
Pre-requirements:
k8s cluster, kubectl, helm



VictoriaMetrics OTel Demo fork

★<https://github.com/VictoriaMetrics-Community/opentelemetry-demo>

The screenshot shows the GitHub repository page for 'opentelemetry-demo' under the 'VictoriaMetrics-Community' organization. The repository has a dark theme. At the top, there's a navigation bar with links for 'README', 'Contributing', and 'License'. Below the navigation, there's a header with the repository name 'OpenTelemetry Demo' and a small icon of a telescope. The repository has a green badge for Slack (@cncf/otel/demo), a purple badge for release v2.1.3, and a blue badge for commits since 2.1.3 (99). It also has a blue badge for docker pulls (639k), a red badge for License (Apache 2.0), and a green badge for Integration Tests (passing). There's a blue badge for Artifact Hub (opentelemetry-demo) and a yellow badge for openssf best practices (in progress 88%). A section titled 'Version for VictoriaMetrics Stack' explains that it's a fork of the OpenTelemetry demo using the VictoriaMetrics Stack, listing three differences: VictoriaMetrics instead of Prometheus, VictoriaLogs instead of OpenSearch, and VictoriaTraces instead of Jaeger. Another section, 'Welcome to the OpenTelemetry Astronomy Shop Demo', describes the repository as a microservice-based distributed system for demonstrating OpenTelemetry. It lists three goals: providing a realistic example of a distributed system, building a base for vendors and tooling authors, and creating a living example for contributors. It also mentions ongoing development and future language support.

KubeCon Atlanta 2025: meeting the community



Keep your on these OpenTelemetry initiatives

[README](#) [Code of conduct](#) [Contributing](#) [Apache-2.0 license](#) [Security](#)

OpenTelemetry Weaver



Observability by Design
Treat your telemetry like a public API

CI [passing](#) [codecov](#) 78% [Security Audit](#) [passing](#) License Apache 2.0 Slack #otel-weaver

OpenTelemetry Weaver helps teams build observability by design, enabling consistent, type-safe, and automated telemetry through semantic conventions. With Weaver, you can define, validate, and evolve your telemetry schemas, ensuring reliability and clarity across your systems.

What is Observability by Design?

Have you ever experienced:

- Broken alerts after a deployment because metric names changed?
- Complex, hard-to-understand queries due to inconsistent naming?
- Teams struggling to interpret unclear or undocumented signals?
- Missing critical instrumentation discovered only in production?

Observability by Design solves these problems by treating your observability signals (metrics, traces, logs) as a first-class public API that requires the same quality standards as your code.

An introduction to Weaver and Observability by Design is presented in the official blog post: [Observability by Design: Unlocking Consistency with OpenTelemetry Weaver](#)

[README](#) [Code of conduct](#) [Contributing](#) [Apache-2.0 license](#) [Security](#)

OpenTelemetry Injector

The OpenTelemetry injector is a shared library (written in [Zig](#)) that is intended to be used via the environment variable `LD_PRELOAD`, the `/etc/ld.so.preload` file, or similar mechanisms to inject environment variables into processes at startup.

It serves two main purposes:

- Inject an OpenTelemetry Auto Instrumentation agent into the process to capture and report distributed traces and metrics to the [OpenTelemetry Collector](#) for supported runtimes.
- Set resource attributes automatically, (for example Kubernetes related resource attributes and service related resource attributes in environments where this is applicable).

The injector can be used to enable automatic zero-touch instrumentation of processes. For this to work, the injector binary needs to be bundled together with the OpenTelemetry auto-instrumentation agents for the target runtimes.

Official RPM and DEB packages that contain the injector as well as the auto-instrumentation agents are available, and can be downloaded from the [releases page](#). The OpenTelemetry injector Debian/RPM packages install the OpenTelemetry auto-instrumentation agents, the `libotelinject.so` shared object library, and a default configuration file to automatically instrument applications and services to capture and report distributed traces and metrics to the [OpenTelemetry Collector](#).

The `opentelemetry-injector` deb/rpm package installs and supports configuration of the following Auto Instrumentation agents:

- [Java](#)
- [Node.js](#)
- [.NET](#)

Support local communities

#otel-docs-localization
#otel-localization-es
#cloud-native-valencia
#cnd-romania
#otel-localization-ro



Inspire other Cloud Native contributors

MOTIVATION = 



VictoriaMetrics

Simple, Reliable, Efficient Monitoring



Robert Klonner  · 1st
DevOps Engineer | Golden Kubestronaut
6d · 

...

Celebrating my first-time Open Source contribution in the ArgoCD Diff Preview project 

Why I am sharing this

I just overcome the initial barrier to do my first OS contribution and I want to motivate others to try it too. It needed several attempts for myself in the last months until it finally worked out. So give it a try and don't give up!

Which strategy finally worked for me

If you are just starting – do not look actively for random projects where you can contribute. Use open source tools and when you stumble upon a problem, raise a well described issue, make suggestions, create documentation you were missing or even improve some lines of code. The benefit of this strategy is that you bring in domain knowledge as you already worked with the tool and probably did some debugging. This fact makes it much easier to make a valuable suggestion.

What brought me to the ArgoCD Diff Preview project

I was looking for a safe way to refactor k8s manifests managed by Kustomize by automatically posting the rendered diff as a pull request comment.
If you also want more insight in manifest changes and use ArgoCD for your deployments I can definitely recommend to try it out

👉 <https://lnkd.in/gt3WpG3k>

What was my contribution

I setup the environment for ArgoCD Diff Preview on Openshift during a POC and had to overcome some issues to make it work. While documenting this for myself I thought I could share that directly with the project and give something back. This resulted in a new documentation section in the project (<https://lnkd.in/gJEE-kU>) and a few lines of Go for a fall back to make the login also work for ArgoCD instances managed by the ArgoCD Operator (<https://lnkd.in/guW8t5Ft>).

Special thanks to:

- **Dag Bjerre Andersen:** For the warm welcome to the project, your super fast reactions, paying attention to all kinds of issues and the straightforward discussions.
- **Kostis Kapelonis:** For the discussion on k8s manifest refactoring strategies and recommending to try ArgoCD Diff Preview.
- **Diana Todea:** For sharing your open source contribution story at the [Cloud Native Days Austria](#) and spreading the motivation to others.

victoriametrics.com

Make your own journey!

- Your motivation is your super power!
- Diversify your contributions
-  code-no-code contributions
- Pair up and mentor
- Share your end user stories!



Resources

[Full-Stack Observability with VictoriaMetrics in the OTel Demo](#)

<https://opentelemetry.io/community/>

<https://training.linuxfoundation.org/certification/opentelemetry-certified-associate-otca/>

<https://github.com/open-telemetry/opentelemetry-injector>

<https://github.com/open-telemetry/weaver>

<https://play.victoriametrics.com/>

<https://play-vmlogs.victoriametrics.com/>

<https://play-vtraces.victoriametrics.com/>

<https://play-grafana.victoriametrics.com/dashboards>

<https://docs.victoriametrics.com/guides/getting-started-with-opentelemetry/>

<https://github.com/VictoriaMetrics-Community/opentelemetry-demo>



Explore VictoriaMetrics Solutions



VictoriaMetrics

Open Source Available on Github



VictoriaTraces

Open Source Available on Github



VictoriaLogs

Open Source Available on Github



Muchas gracias!

Bsky: @didiviking.bsky.social

X: @dianavtodea

Github: @didiViking/Conferences_Talks

LinkedIn: @diana-todea-b2a79968



VictoriaMetrics Community