sumo logic | Continuous Intelligence Platform™

E-BOOK

Monitoring with Prometheus vs. Grafana:

Understanding the difference

Observability has become one of the most important areas of your application and infrastructure landscape, and the market has an abundance of tools available that seem to do what you need. In reality, however, most products — especially leading open-source based products — were created to solve a single problem extremely well, and have added additional supporting functionality to become a more robust solution. However, the non-core functionality is rarely best of breed. Examples of these are Prometheus and Grafana.

What is Grafana?

The one-sentence description right from the source is: "The Grafana project was started by Torkel Ödegaard in 2014 and … allows you to query, visualize and alert on metrics and logs no matter where they are stored."

Essentially, Grafana is a tool that compiles and visualizes data through dashboards from the data sources available throughout an organization. The dashboards handle basic alerting functionality that generates visual alarms. Grafana works best with time-series data, which is what most monitoring and observability platforms produce and store in databases like Graphite, Elastic, or Prometheus's native repository. Through the use of plug-ins, Grafana can also pull data directly from a wide variety of data sources from public cloud providers' monitoring solutions, including Google's Stackdriver and AWS CloudWatch, to SQL databases, like MariaDB and PostgreSQL.

The most commonly mentioned competitor to Grafana is Kibana from the Elasticsearch ecosystem. Kibana and Grafana have the same goal of making it easy to visualize and alert on the data that is available to them — which is also Kibana's biggest weakness. Kibana only supports Elastic as a datasource, while Grafana is not limited to one source.

Kibana's advantage over Grafana is its search capabilities, which makes sense, as it is the tool that Elastic uses in its commercial offering. Extensive search and event correlation are features that only commercial offerings have the time and resources to do well.

Grafana's limitation is that it doesn't have a native capability to aggregate data from multiple sources as it isn't a data store of its own which leads to limited ability to handle correlation across multiple data types.

Getting started with Grafana can be as easy as running a single Docker container and connecting to the Grafana Dashboard.

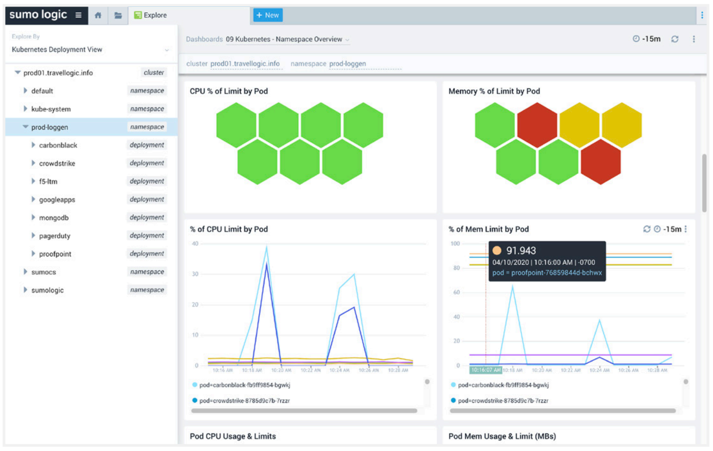
What is Prometheus?

Directly from the source, "Prometheus is an open-source systems monitoring and alerting toolkit…"

That statement is accurate, but does not really address the scale to which Prometheus has caught on as the defacto open-source tool for gathering metrics and generating basic alerts in today's cloud-based world. This is especially true if you are in the Kubernetes universe where it is an undisputed fact that Prometheus is King.

Prometheus has its own datastore to collect the time-series data it generates from the metrics it monitors. Prometheus also has an extensive series of plugins available that allow it to expose data to various external solutions, and to import data from any number of other data sources, including multiple public cloud-monitoring solutions. AWS even recommends Prometheus for its EKS (Kubernetes) offering, over its own CloudWatch service.

Prometheus has data management limitations as it scales. So, instance scalability is dependent on how data is stored, which can be difficult to aggregate back to a holistic view of the infrastructure. For example, when Prometheus hits it these data limits, the administrator has two options to address it: 1) Shard the data to handle the volume using a series of slaves; or 2) segment Prometheus into multiple independent instances. The latter option requires extra tooling to get the holistic view back. A better and third option is to federate the instance through a commercial, platform solution, like Sumo Logic, so scalability issues are masked behind the scenes, producing a consistent holistic view for the administrator with no extra tooling.



A side note: If the series of slave deployment model (with its inherent deployment complexity), is used to address scalability needs, it also resolves an inherent data persistent issue in situations where Prometheus prefers to use local storage. This preference for local storage means that, for most Prometheus deployments, all current and historical data on a node will be lost if a node crashes permanently.

Prometheus has basic visualization capabilities that can be used to expose a small handful of metrics to see basic trending; but almost all organizations expose the data to a more powerful visualization suite.

Docker containers can also run Prometheus. However, the containers are not usually deployed as a standalone. It is most often used within a Kubernetes cluster and is deployed using a Helm chart or managed by an operator. Those two deployment methods take care of a lot of the complexities inherent with running in a Kubernetes cluster and let Prometheus stick to what it is best at which is exposing and gathering the metrics from pods in the cluster.

Versus or better together?

As much as we like to have a single solution to solve every problem, the more complex the infrastructure, the more complex the toolset to support it usually becomes. This is the case with Grafana and Prometheus.

Prometheus focuses on metrics; not logs. It is great at exposing standard and custom metrics from an application it is monitoring. When it is deployed in a Kubernetes cluster it can discover any pod that is running, and persist any time-series data the application has exposed to its data store. Grafana, on the other hand, cannot define what data is exposed and captured.

When Grafana has access to an aggregated data set, it makes it relatively easy to visualize multiple metrics across multiple application stacks on the same screen, in a dashboard that

you can save and refer back to often. Prometheus can visualize individual metrics as graphs, but does not have the same flexibility or extendability as Grafana. Prometheus even links to Grafana in its documentation around visualization, as it knows it has limitations.

Together they make a very powerful combination that covers data collection, basic alerting, and visualization.

Using Grafana with Prometheus is only a few clicks away: simply click "Add New" under datasources in the Grafana console, and enter the connecting information for the Prometheus instance you want to access the data in.

Conclusion

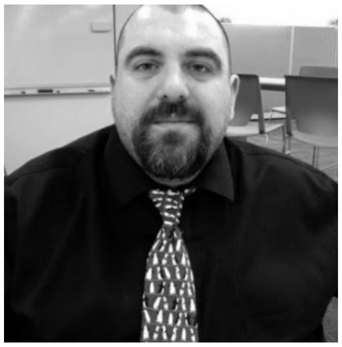
Both Prometheus and Grafana are built around time-series data — with Prometheus primarily on the gathering side and Grafana on the reporting side. Both tools are open-source, are widely available with lots of community support, and are more than capable of meeting the needs of enterprises, large and small.

The two real caveats are the level of expertise required when building a solution with these open-source products. It will be very much a DIY experience, including when leveraging any of Grafana's premade dashboards. The dashboards need more than basic product expertise to import successfully. This is where observability software-as-a-service solutions really show their value. They improve the time-to-value by having premade dashboards readily available, and only a few clicks away — in addition to having a truly centralized data store that consolidates data from all parts of your infrastructure, not one data store per cluster.

The second caveat is the lack of complex alerting logic and even basic event tracking, which can both be accomplished by most commercial offerings. Even the team behind Grafana has a commercial offering with more capabilities around these features.

Learn more about scaling Prometheus here.

About Vince Power



Vince Power is a Solution Architect who has a focus on cloud adoption and technology implementations using open source-based technologies. He has extensive experience with core computing and networking (IaaS), identity and access management (IAM), application platforms (PaaS), and continuous delivery.

About Sumo Logic

Sumo Logic is a leader in continuous intelligence, a new category of software to address the data challenges and opportunities presented by digital transformation, modern applications, and cloud computing. The Sumo Logic Continuous Intelligence Platform™ automates the collection, ingestion, and analysis of application, infrastructure, security, and IoT data to derive actionable insights. For more information, go to sumologic.com.

s u

Continuous Intelligence Platform™

m o

s u
mo

sumo logic

Toll-Free: 1.855.LOG.SUMO | **Int'l:** 1.650.810.8700
305 Main Street, Redwood City, CA 94603

www.sumologic.com