Choose the Tools for Storing and Querying Metrics and Alerting

This lesson focuses on giving a short Introduction to Prometheus and how it can help us to store and query the Metrics, along with sending alerts.

WE'LL COVER THE FOLLOWING

`HorizontalPodAutoscaler (HPA)` and `Cluster Autoscaler (CA)` provide essential, yet very rudimentary mechanisms to scale our Pods and clusters. While they do scaling decently well, they do not solve our need to be alerted when there's something wrong, nor do they provide enough information required to find the cause of an issue. We'll need to expand our setup with additional tools that will allow us to store and query metrics as well as to receive notifications when there is an issue.

Store and query metrics using `Prometheus`

If we focus on tools that we can install and manage ourselves, there is very little doubt about what to use. If we look at the list of Cloud Native Computing Foundation (CNCF) projects, only two graduated in October 2018. Those are **Kubernetes** and Prometheus. Given that we are looking for a tool that will allow us to store and query metrics and that `Prometheus` fulfills that need, the choice is straightforward. That is not to say that there are no other similar tools worth considering. There are, but they are all service-based. We might explore them later but, for now, we're focused on those that we can run inside our cluster. So, we'll add `Prometheus` to the mix.

What is `Prometheus`?

> `Prometheus` is a database (of sorts) designed to fetch (pull) and store highly dimensional time series data.

Time series are identified by a metric name and a set of key-value pairs. Data is stored both in memory and on disk. Former allows fast retrieval of information, while the latter exists for fault tolerance.

How `Prometheus`'s works?

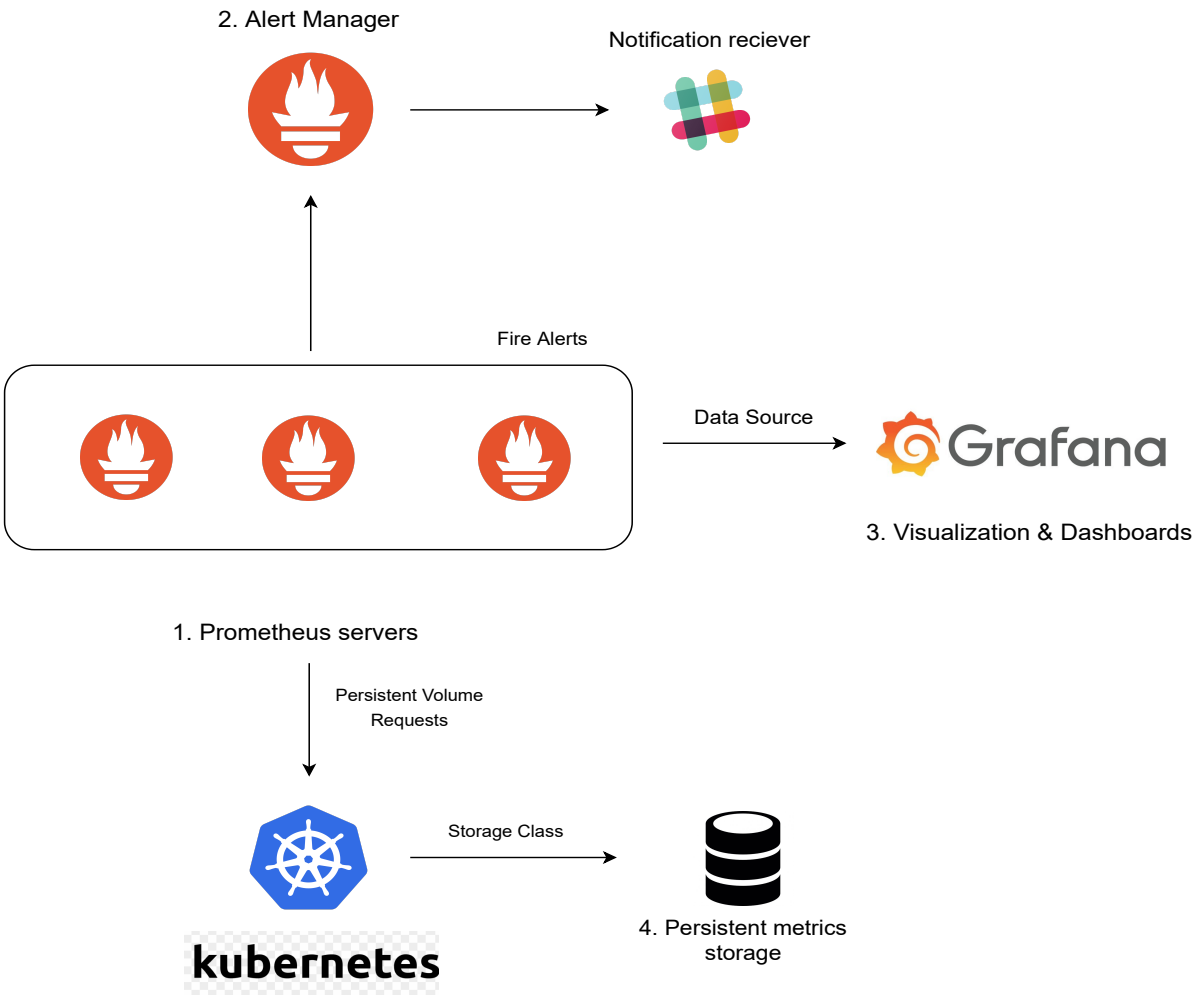
`Prometheus'` query language allows us to easily find data that can be used both for graphs and, more importantly, for alerting. It does not attempt to provide a “great” visualization experience. For that, it integrates with Grafana.

Unlike most other similar tools, we do not push data to `Prometheus`. Or, to be more precise, that is not the common way of getting metrics. Instead, `Prometheus` is a pull-based system that periodically fetches metrics from exporters. There are many third-party exporters we can use. But, in our case, the most crucial exporter is baked into **Kubernetes**. `Prometheus` can pull data from an exporter that transforms information from Kube API. Through it, we can fetch (almost) everything we might need. Or, at least, that’s where the bulk of the information will be coming from.

Send alerts using `AlertManager`

Finally, storing metrics in `Prometheus` would not be of much use if we are not notified when there’s something wrong. Even when we do integrate `Prometheus` with Grafana, that will only provide us with dashboards. I assume that you have better things to do than to stare at colorful graphs. So, we’ll need a way to send alerts from `Prometheus` to, let’s say, Slack. Luckily, Alertmanager allows us just that. It is a separate application maintained by the same community.

We’ll see how all those pieces fit together through hands-on exercises. So, let’s get going and install `Prometheus`, `Alertmanager`, and a few other applications.



How Prometheus & Alertmanager works?

In the next two lessons, we will install and see the usage of `Prometheus` and `Alertmanager`.