Prometheus and Graph Alerts, Grafana Notifications, Semaphores

In this lesson, we will see the difference between Prometheus Alerts and Grafana notifications.

WE'LL COVER THE FOLLOWING

The title might be confusing by itself, so let us briefly describe each of the elements mentioned in it.

`Prometheus` Alerts vs `Grafana` notifications

`Prometheus`'s Alerts and `Grafana` notifications serve the same purpose, even though we did not explore the latter. I'll let you learn how `Grafana` notifications work on your own. Who knows? After the discussion that follows you might not even want to spend time with them.

`Grafana` notifications can be forwarded to different recipients in a similar manner as how `Prometheus`'s alerts are forwarded with `Alertmanager`. However, there are a few things that make `Grafana` notifications less appealing.

If we can accomplish the same result with `Prometheus`'s alerts as with `Grafana` alerts, there is a clear advantage with the former. If an alert is fired from `Prometheus`, that means that the rules that caused the alert to fire are also defined in `Prometheus`. As a result, evaluations are happening at the data source, and we are avoiding unnecessary latency between `Grafana` and `Prometheus`. The closer we are to the data source, the better. In case of alerts/notifications, closer means inside `Prometheus`.

Another advantage of defining alerts in `Prometheus` is the fact that it allows us

to do more. For example, there is no equivalent to Prometheus' `for` statement

in `Grafana`. We cannot define a notification that will fire only if the conditions persist for a while. We'd need to resort to non-trivial additions to the queries to accomplish the same. `Alertmanager`, on the other hand, provides more sophisticated ways to filter the alerts, to group them, and to forward only those that match certain criteria. There are many other advantages to defining alerts in `Prometheus` and `Alertmanager` instead of notifications in `Grafana`. But, we won't go into all of them. I'll leave it to you to find all the differences unless you are already convinced to ditch `Grafana` notifications in favor of `Prometheus` alerts and `Alertmanager`.

There is one important reason why you shouldn't dismiss `Grafana` notifications completely. The data source you're using might not have an alert/notifications mechanism, or it might be part of an enterprise license you do not possess. Since `Grafana` supports many different data sources, with `Prometheus` being only one of them, `Grafana` notifications allow us to use any of those data sources, or even to combine them.

> 🔍 Stick with Prometheus for alerts/notifications based on metrics stored there. For other data sources, Grafana alerts might be better or even the only option. Now that we briefly explored the differences between `Prometheus` Alerts and `Grafana` notifications, we'll move into `semaphores`.

`Semaphores` provide awareness of the System

`Semaphores` (`Grafana` dashboards based on single stat panels) do not replace `Prometheus` alerts. First of all, it is hard, or even impossible, to create semaphores that turn red only if a value reaches a threshold for some time (e.g., like the `for` statement in `Prometheus`'s alerts). That means that a semaphore might turn red, only to go back to green a few moments later. That is not a cause for action since the problem was resolved automatically short while later. If we would jump every time something turns red in `Grafana`, we'd be in excellent physical shape, but we wouldn't do much useful work.

`Semaphores` are an indication of a possible problem that might not require any intervention. While such false positives should be avoided, it's almost impossible to get rid of them altogether. That means that we should stare at

the screen to see whether a red box continues being red for at least a few

minutes before we act. The primary purpose of `semaphores` is not to provide a notification to a person or a team that should fix the issue. Notifications to Slack, email, and other destinations do that. Semaphores provide awareness of the state of the system.

Finally, we explored alerts defined on graphs. Those are the red lines and zones in the graphs. They are not good indicators that there is something wrong. They are not that easy to spot so they cannot raise awareness, and they definitely do not replace notifications. Instead, they help us after we discover that there is an issue. If a notification or a `semaphore` alerts us that there is a problem that might need to be fixed, graph alerts help us identify the culprit. Which Pod is in the red zone? Which ingress received more requests than expected? Those are only a few questions that we can answer through graph alerts.

Q

If an alert is fired from `Prometheus`, that means that the rules that caused the alert to fire are not defined in `Prometheus`.

In the next lesson, we will revise and test the concepts of this chapter through a short quiz.