PROMETHEUS

* Prometheus is an open-source technology designed to provide monitoring and alerting functionality for cloud-native environments, including Kubernetes
* It can collect and store metrics as time-series data, recording information with a timestamp. It can also collect and record labels, which are optional key-value pairs

Key features of Prometheus:

1. **Multidimensional data model:** me and key-value pairs.
2. **PromQL l:A:** flexible querying language that can leverage the multi-dimensional data model.
3. **No reliance on distributed storage:**  All single server nodes remain autonomous.
4. **Pullmodel:** Prometheus can collect time-series data by actively “pulling” data over HTTP.
5. **Pushing time-series data:** Available through the use of an intermediary gateway.
6. **Monitoring target discovery:**  Available through static configuration or service discovery.
7. **visualization:** Prometheus offers multiple types of graphs and dashboards.

### Prometheus for Kubernetes Monitoring:

* Multidimensional data model: The use of key-value pairs creates a similarity to how Kubernetes uses labels to organize infrastructure metadata. This similarity ensures time-series data can be collected and analyzed accurately by Prometheus.
* Accessible format and protocols: Prometheus enables easy and simple exposure of metrics. It ensures metrics are human-readable and can be published via standard HTTP transport.
* **Service discovery:** Prometheus server periodically scrapes targets. Services and applications do not have to constantly emit datametrics are pulled, instead of pushed. Prometheus servers can employ several techniques to auto-discover scrape targets.

Monitor with Prometheus:

Prometheus is a versatile monitoring tool, which you can use to monitor a variety of infrastructure and application metrics. Here are a few common use case.

Host metrics:

You can monitor the operating system to identify when a server’s hard disk is full or if a server operates constantly at 100% CPU. You can install a special exporter on the host to collect the operating system information and publish it to an HTTP-reachable location

Website uptime/up status:

Prometheus doesn’t usually monitor website status, but you can use a blackbox exporter to enable this. You specify the target URL to query an endpoint, and perform an uptime check to receive information such as the website’s response time. You define the hosts to be queried the prometheus.yml configuration file, using relabel\_configs to ensure Prometheus uses the blackbox exporter

Cronjobs:

To check if a cronjob is running at the specified intervals, you can use the Push Gateway to display metrics to Prometheus through an HTTP endpoint. You can push the timestamp of the last successful job (i.e. a backup job) to the Gateway, and compare it with the current time in Prometheus. If the time exceeds the specified threshold, the monitor times out and triggers an alert.

Prometheus monitoring work:

* To get metrics, Prometheus requires an exposed HTTP endpoint
* Once an endpoint is available, Prometheus can start scraping numerical data, capture it as a time series, and store it in a local database suited to time-series data. Prometheus can also be integrated with remote storage repositories.
* Users can leverage queries to create temporary times series from the source
* PromQL can also help you establish alert conditions, resulting in notifications to external systems like email, PagerDuty, or Slack
* Prometheus can display collected data in tabular or graph form, shown in its web-based user interface
* You can also use APIs to integrate with third-party visualization solutions like Grafana.