Observability

Seguimientos, métricas, registros y alertas

Logotipo

Descripción generada automáticamente

Simple Code | Javier



Logotipo, nombre de la empresa

Descripción generada automáticamenteLogotipo, nombre de la empresa

Descripción generada automáticamente

Contenido

[Introducción 2](#_Toc97920881)

[Alertas como código 2](#_Toc97920882)

[Monitoreo de datos con Elasticsearch, Prometheus y Grafana 2](#_Toc97920883)

[Exponer métricas 2](#_Toc97920884)

[Exportadores 2](#_Toc97920885)

[Exportador de Elasticsearch 3](#_Toc97920886)

[Elasticsearch y Kibana 3](#_Toc97920887)

[Protheus y Grafana 3](#_Toc97920888)

[Referencias 4](#_Toc97920889)

# Introducción

La observabilidad es la capacidad de comprender lo que sucede dentro de un sistema a partir del conocimiento de los datos externos que produce, que generalmente son registros, métricas y seguimientos.

La observabilidad se basa en tres pilares: registros, métricas y seguimiento.

# Graylog

Graylog es un estándar de solución de gestión de registros para capturar, almacenar, indexar y permitir el análisis en tiempo real de datos creados para datos estructurados y no estructurados de casi cualquier fuente. Se basa en Elasticsearch , MongoDB y Scala . Graylog se ejecuta en su servidor principal, que recibe datos de casi todos los clientes, que visualiza los datos y permite trabajar con registros estructurados por el servidor principal.

**Configuración de la interfaz web de Graylog**

Los datos que se enviaron al clúster de registro gris no están estructurados. Graylog analiza en formato estructurado y deducible. Mencionaré cómo Graylog hace esto.

Entradas

Los nodos Graylog aceptan datos a través de entradas. Inicie o finalice tantas entradas como desee aquí.

En primer lugar, inicie sesión en la página web de graylog. Si observa el menú superior, verá el botón del sistema que tiene una lista desplegable. Por favor, haga clic en "entradas". Esta página incluye todos los clientes que envían datos desde cualquier fuente (Syslog, gelf (Udp, Tcp, Amqp, Htpp, etc.), cadena sin procesar). Para lanzar una nueva entrada, seleccione el protocolo y el estándar. Es decir, elegimos aquí GELF Udp y hacemos clic en el botón "Iniciar nueva entrada". Verá una ventana emergente en la que puede configurar la entrada GELF.

<https://medium.com/multinetinventiv/integration-graylog-with-gelf-adapter-93b4d5e7c980>

# Alertas como código

<https://www.crowdstrike.com/blog/grafana-alerting-in-a-multi-cloud-world/>

# Monitoreo de datos con Elasticsearch, Prometheus y Grafana

Protheus y Grafana son parte de una plataforma de monitoreo más común en la cultura DevOps basada en microservicios. Prometheus expone un sistema de alertas y recopilación de métricas de temporales. Por otra parte Grafana se complementa con Prometheus siendo una herramienta de visualización.

Prometheus es el responsable de obtener las métricas y supervisa los servicios, este proceso es conocido como scraping.

El servidor Prometheus raspa los puntos finales de servicio definidos, recopila las métricas y las almacena en la base de datos local. Entonces Grafana puede visualizarlos en varios gráficos y tablas. Para consultar las métricas de Prometheus, proporciona un nuevo lenguaje de consulta llamado PromQL. Podemos definir las PromQLconsultas con Grafana para obtener datos de Prometheus.

## Exponer métricas

Para que Prometheus raspe las métricas, cada servicio debe exponer sus métricas (con etiqueta y valor) a través del punto final HTTP /metrics. Por ejemplo, si quiero monitorear un microservicio con Prometheus, puedo recopilar las métricas del servicio (por ejemplo hit count, failure countetc.) y exponerlas con el punto final HTTP. Para hacer eso, necesito crear un servidor HTTP con /metricsun punto final dentro del microservicio. Prometheus viene con diferentes bibliotecas de clientes para hacer eso.

Interfaz de usuario gráfica, Diagrama, Aplicación

Descripción generada automáticamente

## Exportadores

Promethus proporciona una forma de monitorear aplicaciones y servicios de terceros con Exporters. Los exportadores actúan como side-caraplicaciones/servicios de terceros. Recopilan datos de aplicaciones/servicios de terceros y los exponen con un punto final HTTP que Prometheus puede extraer. Hay varios exportadores disponibles con Prometheus. Los exportadores disponibles se pueden encontrar desde [aquí](https://prometheus.io/docs/instrumenting/exporters/) . El exportador más común es [el exportador de nodos](https://github.com/prometheus/node_exporter) , que se puede instalar en cada servidor para leer métricas a nivel del sistema como cpu, etc.memoryfile system

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

# Exportador de Elasticsearch

Para monitorear las métricas de elasticsearch hay un exportador disponible elasticsearch\_exporter. Puede monitorear el clúster health status es de búsqueda elástica index statuses, sharingetc. Puede encontrar más elasticsearch\_exporter información [aquí](https://github.com/justwatchcom/elasticsearch_exporter) . En esta publicación, mostraré cómo visualizar métricas de búsqueda elástica con Prometheus y Grafana usando elasticsearch\_exporter. Todas las implementaciones relacionadas con esta publicación están disponibles en este [repositorio](https://gitlab.com/rahasak-labs/prometheus-elasticsearch) . Clónelo y siga los pasos a continuación.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

# Elasticsearch y Kibana

# Protheus y Grafana

# Referencias

<https://medium.com/rahasak/monitor-elasticsearch-with-prometheus-and-grafana-687a0b6712>