Observability with Traefik

HTTP Forensics for Online-serving systems

The cost of unavailability (opportunity cost)

One of the major negative effects of technology problems is loyalty. These don’t show up in annual reports, but repeated failures – or even one catastrophic event – can erode confidence in a technology company’s brand. And while this is a “soft cost,” decreased loyalty can have a profound effect on a company’s health over time as disappointed users moved to other systems that are perceived as more reliable.

https://blog.rocketsoftware.com/2015/04/the-hidden-costs-of-application-unavailability/

QoS

- Is my chain of services giving a correct response?
- Is my chain of services giving a fast response?
- To (auto-)scale or not to (auto-)scale?

Four golden signals

- Latency
- Errors
- Traffic
- Saturation

https://landing.google.com/sre/sre-book/chapters/monitoring-distributed-systems/

INGRESS

DEAMONSET

/hello

JAEGER

/whoami

FILEBEAT

TRAEFIK

/kibana

/prometheus

ELASTICSEARCH

INGRESS

DEAMONSET

/hello

JAEGER

/whoami

TRAEFIK

/kibana

/prometheus

ELASTICSEARCH

FILEBEAT

Scenarios

- How many requests per second is my service handling?
- How many requests are failing?
- Which requests are so slow that the client is aborting the connection?
- Which requests are within my SLO?
- How does traffic compare with x time ago?
- Which requests never make it to the backend service?
- Good alert threshold?

Questions?

Requests

echo GET $(minikube service traefik --url | head -n 1)/hello | vegeta attack -insecure -rate=100 -format=http | vegeta encode

prometheus (traefik): rate(traefik_service_requests_total{exported_service=~".*whoami.*|.*sb.*"}[1m])

kibana: open "$(minikube service kibana --url)/app/kibana#/discover?_g=()&_a=(columns:!(RequestProtocol,RequestHost,RequestPath,DownstreamStatus,request_Uber-Trace-Id,ServiceName,Overhead,OriginDuration),interval:auto,query:(language:lucene,query:'Duration:+*'),sort:!('@timestamp',desc))"

Errors

echo GET $(minikube service traefik --url | head -n 1)/hello/error | vegeta attack -insecure -rate=1 -format=http | vegeta encode

prometheus (springboot): logback_events_total

Avg request duration

prometheus (traefik): rate(traefik_service_request_duration_seconds_sum[5m]) / rate(traefik_service_request_duration_seconds_count[5m])

Slow requests

http GET $(minikube service traefik --url | head -n 1)/hello/really/slow & http GET $(minikube service traefik --url | head -n 1)/hello/really/slow & http GET $(minikube service traefik --url | head -n 1)/hello/really/slow & http GET $(minikube service traefik --url | head -n 1)/hello/really/slow & http GET $(minikube service traefik --url | head -n 1)/hello/really/slow & http GET $(minikube service traefik --url | head -n 1)/hello/really/slow

jaeger: open $(minikube service jaeger-query --url)

Timeout

echo GET $(minikube service traefik --url | head -n 1)/hello/slow | vegeta attack -insecure -rate=1 -format=http -timeout=300ms | vegeta encode

kibana: open "$(minikube service kibana --url)/app/kibana#/discover?_g=(time:(from:now-24h,mode:quick,to:now))&_a=(columns:!(RequestProtocol,RequestHost,RequestPath,DownstreamStatus,request_Uber-Trace-Id,ServiceName,Overhead,OriginDuration),interval:auto,query:(language:lucene,query:'DownstreamStatus:+499'),sort:!('@timestamp',desc))"

Apdex score

echo GET $(minikube service traefik --url | head -n 1)/hello | vegeta attack -insecure -rate=1 -format=http | vegeta encode & echo GET $(minikube service traefik --url | head -n 1)/hello/slow | vegeta attack -insecure -rate=1 -format=http | vegeta encode

prometheus (traefik): sum(rate(traefik_service_request_duration_seconds_bucket{le="0.3"}[1m])) by (exported_service) / sum(rate(traefik_service_request_duration_seconds_count[1m])) by (exported_service)

CPU / Memory Pressure (SpringBoot Prometheus)

prometheus (springboot): process_cpu_usage

prometheus (springboot): jvm_memory_used_bytes / jvm_memory_max_bytes

Compare Traffic with offset

echo GET $(minikube service traefik --url | head -n 1)/hello | vegeta attack -insecure -h2c -rate=100 -format=http | vegeta encode

prometheus: rate(traefik_service_requests_total{exported_service=~".*whoami.*|.*sb.*"}[5m]) / rate(traefik_service_requests_total{exported_service=~".*whoami.*|.*sb.*"}[5m] offset 5m)

kibana: open "$(minikube service kibana --url)/app/kibana#/discover?_g=()&_a=(columns:!(RequestProtocol,RequestHost,RequestPath,DownstreamStatus,request_Uber-Trace-Id,ServiceName,Overhead,OriginDuration),interval:auto,query:(language:lucene,query:'Duration:+*'),sort:!('@timestamp',desc))"