INSTRUMENTATION DEMO

July 24, 2024

GET THE SOURCE

- git clone https://github.com/retzkek/myjob.git
- Requires Go 1.22
  - brew install go
  - apt-get install golang
  - dnf install golang
  - nix-shell
  - https://go.dev/dl/
  - podman build –t myjob .
    - Builds binary in container.

REPOSITORY FORMAT

- Each step of the demo is in a branch:
  - 01-server
  - 02-logging
  - 03-metrics
  - 04-tracing
- git checkout 01-server

BUILDING AND RUNNING

- Go installed:
  - export LENS_URL=https://landscape.fnal.gov/lens/query
  - go build && ./myjob
- Docker/podman:
  - podman build -t myjob .
  - podman run -it --rm -p 127.0.0.1:8888:8888 \
    –e LENS_URL=https://landscape.fnal.gov/lens/query \
    myjob -a 0.0.0.0:8888

STEP 01: BASIC SERVER

`git checkout 01-server`

- Accepts requests at /status/<jobid> and gets job info from the Lens API
- Submit job with jobsub
- Get its status
  - curl -i localhost:8888/status/<jobid>
- Don’t worry too much about the Lens client (for now, feel free to investigate later)

STEP 02: ADD LOGGING

`git checkout 02-logging`

- Add logrus library for structured logging, and some basic request logging
- We could use filebeat, fluentd, fluentbit, or promtail to collect the logs and ship them to Landscape
- If you run your application in OKD, it already sends the logs to Landscape!
  - https://landscape.fnal.gov/monitor/d/JJQVkz74z/okd4-container-logs?orgId=1

STEP 03: ADD METRICS

`git checkout 03-metrics`

- Add the Prometheus client library and register a histogram metric of request durations
- Could point Prometheus at the /metrics endpoint to collect these metrics

STEP 04: ADD TRACING

`git checkout 04-tracing`

- Add the OpenTelemetry library and initialize the tracing module
  - Yeah, this is a bit of a doozy, with a fair bit of boilerplate.
- Important:
  - set serviceName to something unique for yourself (or export OTEL_SERVICE_NAME env var)
  - or set OTEL_RESOURCE_ATTRIBUTES env var to something like "myname=kretzke"
  - also set OTEL_EXPORTER_OTLP_ENDPOINT as per https://landscape.fnal.gov/docs/data/tracing/
- We log the trace id, which you can copy-paste into Jaeger or Grafana to view the trace
  - https://landscape.fnal.gov/jaeger/search
  - https://landscape.fnal.gov/monitor/explore

NEXT STEPS

- Why doesn’t the trace show the Lens API call and connect to the Lens trace?
  - Lens uses Jaeger and OpenTracing libraries instead of OpenTelemetry. Should be compatible, but…
- Set up a dashboard with an alert on the request duration metric to let us know if we’re not meeting SLA (or some ”reasonable” threshold that indicates someone should check on the service)
- Set up blackbox monitoring of our service to make sure it is responding.
- Deploy to OKD