DrDroid

OpenTelemetry Collector Cheatsheet

Installation

curl -sSL https://raw.githubusercontent.com/open-telemetry/opentelemetry-collector/main/examples/local/otel-config.yaml -o otel-config.yaml
Downloads a basic OpenTelemetry Collector configuration file

docker run -p 4317:4317 -p 4318:4318 -v $(pwd)/otel-config.yaml:/etc/otel-config.yaml otel/opentelemetry-collector --config=/etc/otel-config.yaml
Runs OpenTelemetry Collector in a Docker container with the specified configuration file

brew install opentelemetry-collector
Installs the collector via Homebrew (macOS)

Configuration

otelcol --config=config.yaml
Starts the collector with a specific configuration file

otelcol --config=config.yaml --set=service.telemetry.logs.level=debug
Starts the collector with debug logging enabled

otelcol validate --config=config.yaml
Validates a configuration file without starting the collector

Health Check

curl http://localhost:13133/health
Checks the health status of the collector

curl http://localhost:8888/metrics
Gets the Prometheus metrics from the collector

Pipelines

otelcol --config=config.yaml
Runs a collector with receivers, processors, and exporters defined in config.yaml

docker logs <collector-container-id>
Views logs from a containerized collector

Data Endpoints

curl -X POST -H "Content-Type: application/json" http://localhost:4318/v1/traces -d '{"resourceSpans": [...]}'
Sends trace data to the HTTP endpoint

curl -X POST -H "Content-Type: application/json" http://localhost:4318/v1/metrics -d '{"resourceMetrics":[...]}'
Sends metrics data to the HTTP endpoint

curl -X POST -H "Content-Type: application/json" http://localhost:4318/v1/logs -d '{"resourceLogs": [...]}'
Sends logs data to the HTTP endpoint

Common Ports

4317
OTLP gRPC receiver

4318
OTLP HTTP receiver

8888
Prometheus metrics endpoint

13133
Health check endpoint