Observability for Lean Teams

Open source tools and Best practices

- Distributed systems are complex—tracing issues is hard!
- Enterprise solutions are expensive and overkill for small teams.

Objectives:

- Build a minimal observability stack in Kubernetes
- Keep costs low while gaining powerful insights
- Avoiding pitfalls

Pillars of Observability

Pillars of Observability

- Logging: Fluentd (https://github.com/fluent/fluentd)
- Monitoring (Metrics): Prometheus (https://github.com/prometheus/prometheus)
- Tracing: Jaeger (https://github.com/jaegertracing/jaeger)
- Dashboards: Grafana (https://github.com/grafana/grafana)

Logging - Minimal Setup

Docs:

- With Kubernetes Daemonset:

```
kubectl apply -f
https://raw.githubusercontent.com/fluent/fluentd-kubernetes-daemonset/master/fluentd-daemonset-elasticsearch-rbac.yaml
```

- With Helm:

```
helm repo add fluent https://fluent.github.io/helm-charts
helm repo update

helm install fluentd fluent/fluentd
```

- Apps can push logs to Fluentd TCP/UDP ports (24224) using the Fluentd forward protocol.

- Best practice: Use well structured JSON logs for easy parsing.

Metrics - Minimal Setup

Docs:

- With Helm:

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
helm repo update

helm install prometheus prometheus-community/prometheus
```

- Apps can use client libraries to push to export `/metrics` endpoint
- Override `values.yaml` in helm

```
scrape_configs:
  - job_name: 'my-app'
    static_configs:
      - targets: ['my-app-service:8080']
```

Tracing - Minimal Setup

Docs:

- With Kubernetes:

```
kubectl apply -n observability -f
https://github.com/jaegertracing/jaeger-operator/releases/download/v1.47.0/jaeger-operator.yaml
```

- With Helm:

```
helm repo add jaegertracing https://jaegertracing.github.io/helm-charts
helm repo update

helm install jaeger jaegertracing/jaeger
```

- Apps can use client libraries to export data to Jaeger's collector endpoint in your K8s cluster.

```
jaeger_exporter = JaegerExporter(
   agent_host_name="jaeger-agent",  # Replace with your Jaeger agent dns
   agent_port=6831,  # Replace with your Jaeger agent port
)
```

Dashboard - Minimal Setup

Docs:

- With Kubernetes:

```
kubectl apply -f
https://raw.githubusercontent.com/grafana/helm-charts/main/charts/grafana/templates
/deployment.yaml
```

- With Helm:

```
helm repo add grafana https://grafana.github.io/helm-charts

helm install grafana grafana/grafana
```

- Prometheus, Jaeger can be added as data sources to Grafana.

- Override `values.yaml` in helm

```
datasources:
  - name: Prometheus
    type: prometheus
    access: proxy
    url: http://prometheus-server
```

Best Practices and Lessons learned

- Avoid **alert fatigue**
    - **Stories from the trenches**: “We once had 200+ alerts… and ignored all of them.”
- Don’t oversample metrics: strike balance
- Dashboard hygiene – don’t just dump all panels

Closing and Q&A

- Recap (Pillars of observability):
  - Logging
  - Monitoring (metrics)
  - Tracing
  - Dashboards

Linkedin: https://linkedin.com/in/vgnshiyer

Github: https://github.com/vgnshiyer

Blog: https://blog.vgnshiyer.dev