REST/API Monitoring

Yarden Krupik

15/05/2025

What is REST/API Monitoring?

What is API Monitoring?

What is API?

- Application Programming Interface
- A set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service (Oxford)

What is REST API?

- Representational State Transfer
- Uses GET, POST, PUT and DELETE

What is API Monitoring?

- How fast the API responds
- How much traffic it handles
- Whether it’s working properly

Key Metrics for API Monitoring

- Latency - Time taken to get a response
- Throughput - Number of requests handled per time unit (ms/s)
- Error Rate - Ratio of failed requests

- For our project:
- Request count, Resource Usage, Execution Times, Source and target system metrics.

Popular Tools

- Prometheus
- Grafana
- Grafana Loki
- Zabbix
- Postman Monitoring
- New Relic

Prometheus

- Metrics collector and alerting system
- Pulls metrics from targets via http endpoints
- Open source
- Release: 2012
- Repository: github.com/prometheus/prometheus

Grafana

- Visualization and dashboard tool
- Connects to Prometheus to create interactive dashboards
- Release: 2014
- Repository: github.com/grafana/grafana

Grafana Loki

- Log aggregation system that's designed to be cost-effective and easy to operate
- Well-suited for use with Prometheus and Grafana. Connects to Prometheus to create interactive dashboards.
- Release: 2018
- Repository: github.com/grafana/loki

Zabbix

- Enterprise-grade monitoring platform
- Agent-based or agentless data collection via SNMP, HTTP and scripts
- Release: 2001
- Repository: git.zabbix.com/projects/zbx/repos/zabbix/browse

Postman Monitoring

- Cloud-based API and monitoring
- User can schedule API tests (collections) to run from different locations
- Release: 2012

New Relic

- Real-time dashboards, tracing, alerts, more for enterprises with budget.
- Founded: 2008

Comparison overview

Tool	Open Source	Metrics	Dashboards	Alerts	Tracing	Best Use Case
Prometheus	+	+	+ (with Grafana)	+	limited	High-frequency metrics
Grafana	+	+ (visualizes)	+	+	with Tempo	Visualizing metrics
Zabbix	+	+	+ (built in)	+	-	All-In-One monitoring
Postman	- (free tier)	some	+ (basic)	+	-	Functional test monitoring
New Relic	-	+	+	+	+	Enterprise observeability
Grafana Loki	+	-	+(with Grafana)	-	-	Log collection, correlated with metrics

Architecture Setups for Distributed Tracing and Logging

- Distributed Tracing:
  - Trace data synchronization requests across the system. Crucial for identifying bottlenecks or failure points in the data pipeline.
  - Proposed Setup:
    - Instrumentation (OpenTelemetry SDKs)
    - Storage (Jaeger or Tempo)
    - Visualization (Jaeger UI or Grafana/Tempo).

Architecture Setups for Distributed Tracing and Logging

- Distributed Logging:
  - To centralize logs for debugging and reviewing
  - Proposed Setup:
    - Agents/Forwarders (Fluent Bit).
    - Aggregation Layer (Loki).
    - Storage (Loki).
    - Visualization/Querying (Grafana).

Strategies for Scaling and Alerting

- Scaling Strategies:
  - Handle increasing data load and synchronization demands
  - Recommendations:
    - Scale Prometheus and Grafana horizontally
    - Use Prometheus federation or remote storage (like Thanos) for long-term scalability
    - Scale Sync Nodes
    - Consider message queues for distributing synchronization tasks

Strategies for Scaling and Alerting

- Alerting Strategies:
  - Detect and respond to synchronization issues promptly
  - Recommendations:
    - Use Prometheus Alertmanager.
    - Define alerts for key metrics (such as synchronization latency, error rates, resource utilization).
    - Route alerts to appropriate teams based on the source of the issue.
    - Implement alert grouping and deduplication.

conclusion

- This presentation gave an overview of REST/API monitoring for our 'Stay in Sync' project
- Monitoring is important for data consistency, troubleshooting, and system health.
- Tools like Prometheus, Grafana, and Loki help with metrics, visualization, and logs.
- Tracing and logging provide detailed insights for optimization.
- Scaling and alerting strategies ensure the system is resilient and responsive.

Sources

- prometheus.io
- grafana.com
- grafana.com/oss/loki
- opentelemetry.io
- Wikipedia en.wikipedia.org/
- fluentbit.io
- jaegertracing.io