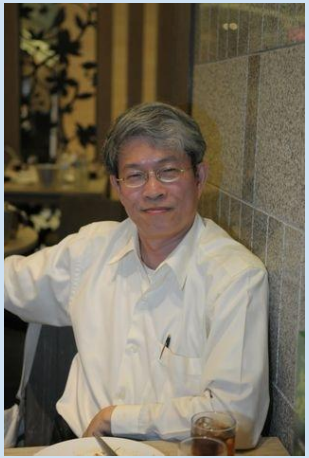
Observability



Dr. Yunyong Teng-amnuay

(Retired)

Dept. of Computer Engineering

Faculty of Engineering

Chulalongkorn University

November 7, 2024

What is observability?

https://www.ibm.com/topics/observability#:~:text=Observability%20is%20the%20extent%20you,knowledge%20of%20its%20external%20outputs.

- Observability is the extent you can understand the internal state or condition of a complex system based only on knowledge of its external outputs.
- The more observable a system, the more quickly and accurately you can navigate from an identified performance problem to its root cause, without additional testing or coding.
- Observability provides deep visibility into modern distributed applications for faster, automated problem identification and resolution.

What is observability?

https://www.ibm.com/topics/observability#:~:text=Observability%20is%20the%20extent%20you,knowledge%20of%20its%20external%20outputs.

A relatively new IT topic, observability is often mischaracterized as an overhyped buzzword, or a “rebranding” of system monitoring, application performance monitoring (APM), and network performance management (NPM). In fact, observability is a natural evolution of APM and NPM data collection methods that better addresses the increasingly rapid, distributed and dynamic nature of cloud-native application deployments.

Why observability?

https://www.ibm.com/topics/observability#:~:text=Observability%20is%20the%20extent%20you,knowledge%20of%20its%20external%20outputs.

- Modern development practices: agile development, continuous integration and continuous deployment (CI/CD), DevOps, multiple programming languages.
- Cloud-native technologies: microservices, Docker containers, Kubernetes and serverless functions.
- Bringing more services to market faster, components, locations, languages, in seconds.
- Need higher quality telemetry for a high-fidelity, context-rich, fully correlated record of every application user request or transaction.

Components of observability

https://www.ibm.com/topics/observability#:~:text=Observability%20is%20the%20extent%20you,knowledge%20of%20its%20external%20outputs.

- **Logs**: timestamped, immutable records of application events, high-fidelity, millisecond-by-millisecond record of every event. Developers can 'play back' for troubleshooting and debugging.
- **Metrics**: measures of application and system health, for example, how much memory an application uses in five-minutes, or how much latency an application experiences during a spike in usage.
- **Traces**: end-to-end 'journey' of user request, from the UI or mobile app through the entire distributed architecture and back to the user.
- **Dependencies**: reveal how each application component depends on other components, applications and IT resources.

Benefits of Observability

https://www.ibm.com/topics/observability#:~:text=Observability%20is%20the%20extent%20you,knowledge%20of%20its%20external%20outputs.

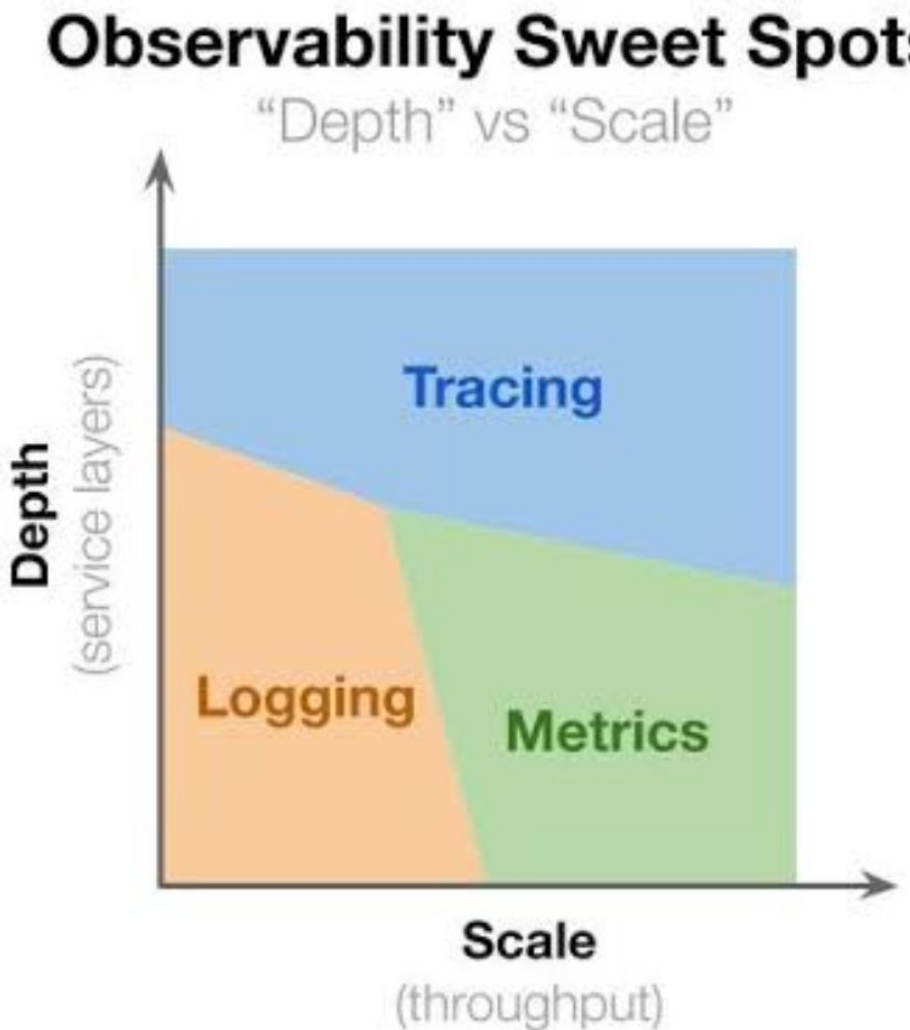
- **Discover 'unknown unknowns'**–issues you don't know exist.
- **Catch and resolve issues early**. DevOps teams can identify and fix issues in new code before they impact the customer experience.
- **Scale observability automatically**. Specify instrumentation and data aggregation as part of a Kubernetes cluster configuration and start gathering telemetry from the moment it spins up, until it spins down.
- **Automated remediation and self-healing**. Combine observability with AIOps machine learning and automation capabilities to predict issues and resolved them without management intervention.

Fragmented Tools

- Multicloud app employs 12 platforms & services on the average.
- Organization uses 10 different observability/monitoring tools.
- Too much time to maintain tools and prepare data for analysis.
- Kubernetes scales services through dynamic resource provisioning.
- Difficult to maintain visibility in Kubernetes environment.
- Log analytics is a challenge. Storage vs. query.

Solutions

- Unified platform for observability and security.
- AIOps - AI for IT operations, combining multiple AI techniques.



Sweet Spots on Data Usage

- Each type of data (traces, logs, metrics) has its own use.
- Do not persist on a single type. Depends on use case.
- Example, extending logs to other use cases will result in lower performance.

Open source observability explained - Grafana Labs stack

Senior Developer Advocate Nicole van der Hoeven

https://www.youtube.com/watch?v=WSW1urIXsfA&t=229s

TIMESTAMPS

00:00 Intro
00:31 Concerns in observability
01:05 1. Identifying data to collect
02:23 Logs with Grafana Loki
04:16 Metrics with Prometheus and Grafana Mimir
06:26 Distributed tracing with Grafana Tempo
08:00 Continuous profiles with Grafana Pyroscope
10:16 2. Collecting data and instrumentation
10:50 Source instrumentation with Grafana Faro and OpenTelemetry
11:32 Binary instrumentation with Grafana Alloy
12:52 External eBPF instrumentation with Grafana Beyla
14:24 Software testing with Grafana k6
16:09 3. Doing stuff with data - visualization with Grafana
17:22 Incident response management with Grafana OnCall
18:27 Summary of open source observability

KEYWORDS

1. Telemetry
2. Logs
3. Metrics
4. Traces
5. Profiles
6. Optimization
7. Scalable
8. Multitenant
9. Instances
10. Query language
11. Parsing
12. Ephemeral
13. Aggregation
14. LOKI
15. Metadata
16. Prometheus
17. Troubleshooting
18. Microservices
19. Kubernetes
20. PROMQL
21. Exemplar
22. Mimir
23. Transaction
24. Instrumented
25. TEMPO
26. TraceQL
27. PromQL
28. PYROSCOPE
29. Visualization
30. Flame graphs
31. FlameQL
32. Instrumentation
33. FARO
34. SDK
35. Alloy
36. BEYLA
37. EBPF
38. gRPC
39. K6
40. ONCALL

Q & A

Thank You !!

12