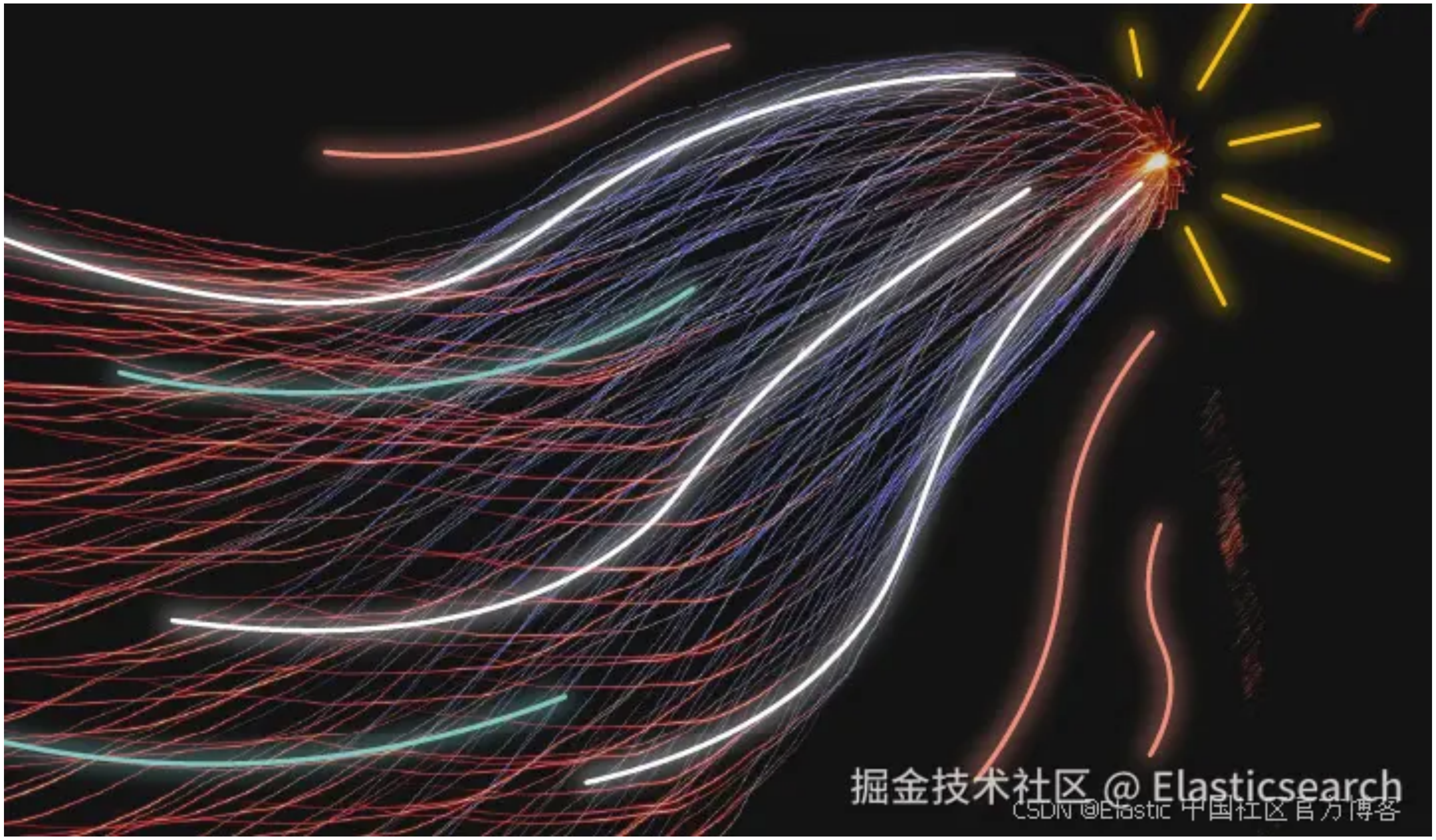
Understanding observability metrics: types, golden signals, and best practices

Elasticsearch 2025-03-31 75 9-minute read 关注

By: Elastic Elastic Observability Team



Observability metrics provide insights into the performance, behavior, and health of applications, systems, and infrastructure—enabling the practice of observability, which is understanding the internal state of a system by inspecting data. As organizations continue to collect more and more data, observability metrics are important telemetry signals for observability .

In modern application development, observability refers to the collection and analysis of telemetry data from various sources— logs , metrics , and traces —to gain insight into the behavior of applications running in an environment. Observability metrics are telemetry signals that help organizations understand their operations and create proactive monitoring processes.

By leveraging observability metrics, organizations can gain a comprehensive understanding of the performance of their technology stack, improving problem diagnosis and resolution time. When used effectively, observability metrics can provide valuable business insights, drive growth, and enable organizations to focus on innovation.

The three pillars of observability

The foundation of observability is often described as three pillars: metrics, logs, and tracing. Together, they provide critical visibility into system performance and behavior. As technology continues to advance and the demand for observability increases, a fourth pillar is emerging: profiles.

index

Metrics are raw, numeric data points collected from hardware, software, and websites. Metrics are used to monitor resource usage, performance, and user behavior by measuring known knowns. In other words, metrics tell monitoring and observability teams what's happening in their systems.

Core types of observability metrics
Observability is a practice that provides organizations with a 360-degree view of their environment and operations. To do this, observability relies on these core types of metrics:

- **Application Metrics** : Application metrics are telemetry data generated by applications and related to the applications in the technology stack. Some commonly used metrics include response time, throughput, request rate, and error count. These metrics enable engineers to monitor application performance and availability. Application metrics are also used in application performance monitoring (APM) .

- **System metrics** : System metrics, also known as infrastructure metrics, reflect the health of the hardware and operating system, including critical components like Kubernetes . Examples include CPU utilization, disk I/O, network throughput, memory usage, instance uptime, container resource usage, and service availability. These metrics provide insight into the performance of cloud resources, virtual machines, containers, and other underlying components.

- **Business metrics** : Business metrics tie technical and operational performance to business results. For example, metrics like conversion rate, average transaction value, and user retention help tie system performance to organizational goals.

An effective observability solution ensures reliability, efficient resource allocation, compliance, and security. It also helps plan capacity, optimize performance, improve user experience, and control costs. Core metrics enable effective observability and ultimately enable data-driven decision-making, leading to better business outcomes. These metrics are often aggregated and visualized in dashboards for real-time performance monitoring.

log

Logs are time-stamped entries of specific events generated by systems, applications, networks, and infrastructure. They provide details and context about events, helping engineers understand why problems occurred.

Network devices, applications, operating systems, IoT devices, and third-party applications emit different types of logs, including (but not limited to):

- **System log** : Includes events such as connection attempts, errors, and configuration changes.

- **Application logs** : Record software changes, CRUD operations, application authentication, and other events to help diagnose problems.

- **Network logs** : Data that records events that occur on a network or device, including network traffic, security events, and user activity.

Logs are recorded in structured and unstructured formats, which presents a storage challenge. Log data can also be difficult to classify because it is often siloed across various systems and not automatically correlated.

track

Traces are telemetry signals that allow engineers to view applications and services from the perspective of a user session. Distributed tracing collects trace data about requests that traverse a distributed architecture.

Traces allow engineers to monitor and debug applications and identify bottlenecks. In other words, traces tell DevOps teams where problems are occurring. They are the foundation of proactive monitoring. By analyzing traces, engineers can identify metrics or logs related to specific issues, thereby mitigating future problems.

For example, traces that help identify slow processes include API queries, front-end API traffic, workload between servers, and internal API calls.

While metrics, logs, and traces provide users with valuable data about application and system performance, these signals don't always provide the detailed information needed to troubleshoot code issues and optimize performance. This is where analytics comes into play.

Analysis - profiles

Profiling is the collection and analysis of stack traces that helps identify issues related to data structures, code visibility, and memory allocation, both at the kernel and user levels.

Analytics helps identify bottlenecks in the system at the code level, which is another key benefit of modern observability. OpenTelemetry is also using analytics as a signal. As a result, analytics is becoming the fourth and newest pillar of observability.

Key Observability Indicators: Four Golden Signals for SRE Teams

While every organization's monitoring needs are unique, certain observability metrics are universally important. These are sometimes referred to as the four golden signals in the site reliability engineering (SRE) community.

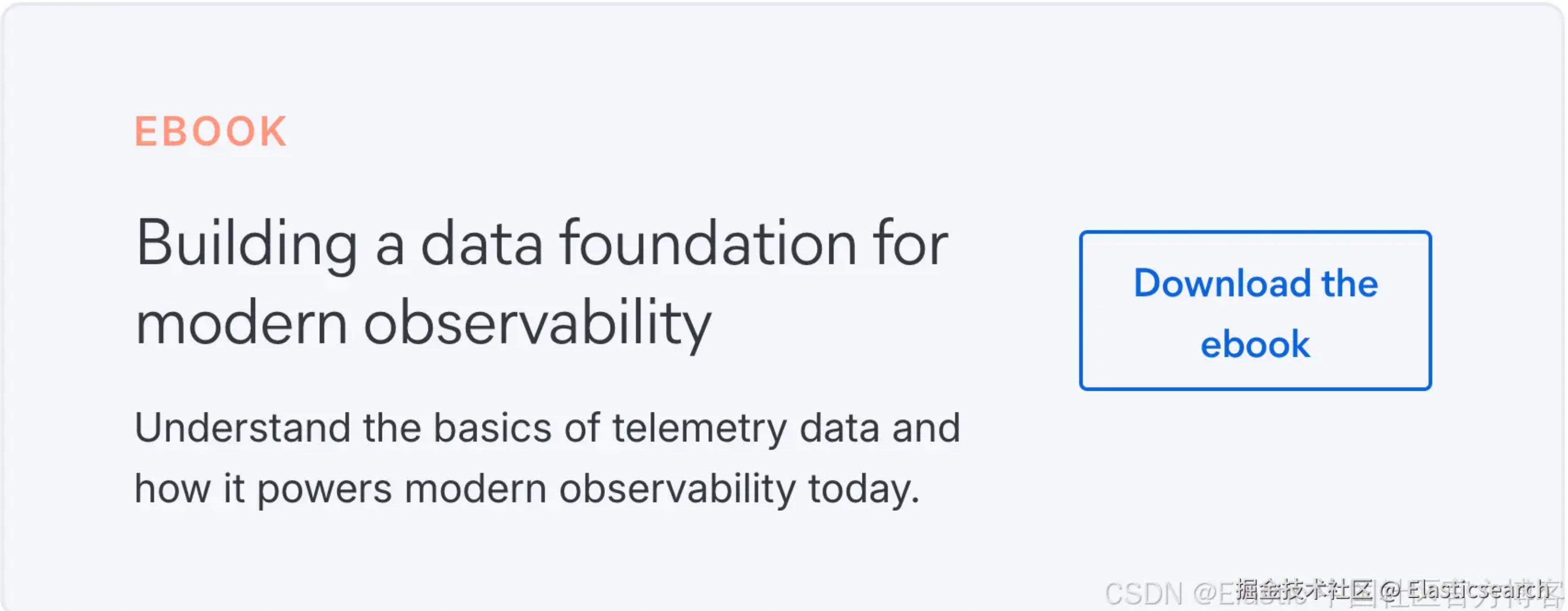
- **Delay**
  Latency measures the time it takes for data to travel from one point to another. Latency can indicate potential performance issues. High latency can degrade the user experience by increasing load times, causing application errors, and challenging user expectations.

- **flow**
  Traffic metrics track the number of requests or transactions processed by an application. They help teams understand user behavior and predict scaling needs.

- **mistake**
  Error metrics provide visibility into failed requests or operations. Monitoring error rates and identifying patterns can help resolve recurring issues.

- **Saturation**
  Saturation metrics indicate how close a system is to reaching its capacity limits. Monitoring resource utilization ensures engineers can proactively address bottlenecks before performance is impacted.

These four golden signals are key to effective observability practices because they provide insights into the health and performance of IT systems. When these metrics are monitored, correlated, and analyzed, they help IT teams gain actionable insights, enabling them to take a more proactive stance on site reliability and performance monitoring.



Download eBook

Best practices for implementing observability metrics

The primary challenge in implementing observability metrics is sifting through the noise—many signals generate a large volume of telemetry data , not all of which is useful. Furthermore, SRE teams often face the problem of data heterogeneity. How can you correlate these various types of data to make troubleshooting easier?

From these challenges, we can identify some best practices for implementing observability metrics.

- **Define clear goals** : Successfully implementing observability metrics—and combating data overload—begins with identifying your goals. When defining these goals, ask yourself what you need these metrics to tell you. You don't need to monitor everything; you only need to monitor what's critical to your organization and systems.

- **Use open standards for application monitoring** : Monitoring is the process of generating and collecting telemetry data from your applications. To avoid vendor lock-in when adding applications, consider using a vendor-neutral framework like OpenTelemetry (OTel) . OTel provides a standardized framework that enables you to collect and compare telemetry data from multiple sources.

- **Leverage automation** : Automate data collection, analysis, and alerting to reduce manual efforts and increase response speed.

- **Customize visualizations** : It’s best to customize your dashboards to achieve your defined goals. Default dashboards are only useful up to a certain point — customizing how you visualize your environment is key to successful observability.

Using Elastic's observability metrics

Elastic Observability provides a unified solution for collecting, monitoring, and analyzing observability metrics across your entire technology stack. With Elastic Observability, you can collect, store, and visualize observability metrics from any source and with our Search AI platform . accelerate problem resolution

Elastic Observability helps accelerate problem resolution through search-based relevance, downtime prevention, uncompromised data retention, improved operational efficiency and reduced costs, and future-proof investment. With an open, OTeL-first solution, you can gain fast, contextual, and unified insights across the broadest range of data sources, seamlessly integrating with an evolving technology ecosystem.

Learn more about observability at Elastic .

The release and timing of any features or functionality described in this document remains entirely at the discretion of Elastic. Any features that are not currently available may not be delivered on time or at all.

原文： Understanding observability metrics: Types, golden signals, and best practices | Elastic Blog

上一篇 AI 驱动的安全分析的价值是什么?

下一篇 RAG vs. Fine Tuning，一种实用方法

登录 / 注册　即可发布评论!

暂无评论数据

Table of contents

Close

Using Elastic's observability metrics

Related recommendations

Featured Content

Ten core advantages of Maven multi-module projects

zjjuejin · 49 reads · 1 like

Rust Practical Advanced: In-depth Analysis of the Mystery of Rust Lifecycle

Xunyue Yinjun · 39 reads · 0 likes

为你推荐

Observability：什么是可观察性?

Elasticsearch 4月前 159 点赞 评论 Elasticsearch

Observability：了解可观察性的启蒙书

Elasticsearch 3年前 347 点赞 评论 后端

可观察性入门

蚂蚁背大象 2年前 1.4k 3 7 后端 云原生 掘金·金石...

深度解析丨基于 eBPF 的 Kubernetes 一站式可观测性系统

阿里云云原生 3年前 1.9k 1 评论 Kubernetes

READS: Salesforce服务健康指标最佳实践

俞凡 3年前 384 1 1 微服务

使用 OpenTelemetry 和 SigNoz 实现 LLM 可观测性

threerocks 1年前 1.4k 2 评论 前端 JavaSc... 人工智能

中小企业的 Kubernetes 最佳实践（二）：应对可观察性的挑战

DigitalOcean 11月前 73 点赞 评论 Kubernet... 运维

可观察性成熟度模型

方石剑 2年前 464 点赞 评论 后端

深度解析丨基于 eBPF 的 Kubernetes 一站式可观测性系统

阿里云云栖号 3年前 1.3k 1 评论 Kubern...

提升系统管理：监控和可观察性在DevOps中的作用

陈哥聊测试 1年前 823 1 评论 DevOps

流利说统一可观察性平台实践

阿里云云栖号 4年前 690 1 评论 运维

重新思考可观测性

云云众生s 1年前 29 点赞 评论 监控

Observability：可观察性的 4 个好处

Elasticsearch 12月前 61 点赞 评论 Elasticsearch

Elastic：如何实现业务可观测性

Elasticsearch 4月前 52 点赞 评论 Elasticsearch

阿里可观测性数据引擎的技术实践

阿里云云栖号 3年前 2.5k 2 评论 阿里巴巴