Data for Breakfast Around the World | Drive impact across your organization with data and agentic intelligence. Register Now

WHAT IS OBSERVABILITY? FULL GUIDE 2025

Explore observability, its benefits, examples and use cases. Learn how it boosts system performance, reliability and analytics.

- Overview
- What Is Observability?
- What Is the Difference Between Observability and Monitoring?
- How Does Observability Work?
- Why Is Observability Important for Your Business?
- The Pillars of Observability
- Benefits of Implementing Observability in Your Organization
- How Engineers and Developers Benefit from Observability
- Challenges Faced When Implementing Observability
- How to Successfully Implement Observability in Your Environment
- Three Features to Look for in an Observability Tool
- Conclusion
- Observability FAQs
- Customers Using Snowflake
- Observability Resources

OVERVIEW

Observability has become a critical component of many modern organizations, especially following the widespread adoption of cloud-native systems and software. At its most basic level, observability refers to the ability to assess a system's internal state based on its outputs. But this simple definition belies the high levels of complexity that define most modern systems.

Most organizations generate a large volume of data as part of their operations, whether it be related to marketing performance, platform usage or some other metric. Many of these systems are also distributed across multiple hosting environments, thanks to the advent of scaling tools like containerization and multicloud models. Observability allows organizations to track the operation and performance of all these systems, allowing them to identify and remediate issues, find inefficiencies and get a high-level understanding of your organization's processes.

In addition to helping you proactively address any performance issues, strong observability also makes it easier to add new tools and processes and make your IT and operations teams more productive.

In this article we're going to explore observability in more detail, the tools and strategies that comprise an observability infrastructure and the best ways for you to identify and incorporate observability into your systems.

WHAT IS OBSERVABILITY?

In a modern IT system, observability is the ability to assess the internal performance of your systems based on their outputs. Usually to do this, an enterprise needs a set of tracking tools and functions that provide a single source of truth to assess that performance. Achieving observability usually demands a combination of instrumentation types which combine activity logs, performance metrics, tracing and end user input to provide a holistic assessment of your systems.

Combining multiple approaches is critical, as it allows you to address issues between and within complex systems where a single metric or activity log is not enough. Building an observability system this way doesn't just help monitor for common issues like service interruptions or latency but goes beyond these to identify unexpected problems that can crop up in more complex environments.

WHAT IS THE DIFFERENCE BETWEEN OBSERVABILITY AND MONITORING?

Although they may take similar approaches, observability and monitoring are two different ways to assess a system's performance.

Monitoring: Monitoring uses a set of predefined rules and metrics to measure activity and notify you if there is anything outside of those parameters. For example, if a database's response time doubles, this would alert you that something is wrong with the database configuration or the queries you are using. Conversely, monitoring is a pre-mediated set of measurements designed to catch common or expected issues.

Observability: Observability goes beyond being able to monitor one element of a system. Rather, multiple measurements help assess all the elements and the interactions among them. This allows you to take stock of the performance of a highly complex set of services and identify the root cause of specific issues.

For example, perhaps someone made a change to a database schema which cuts query response time in half, but which also changes the output in a way which breaks downstream systems. An observability approach would help you identify the change which caused this issue and come up with a solution, while monitoring might only tell you that something had broken.

HOW DOES OBSERVABILITY WORK?

Aggregating performance metrics and internal activity logs, tracing data as it moves through a system and combining all of this with end user experience metrics, observability gives you an all-in-one perspective into the performance of that system. Setting up strong observability not only assists in identifying the source of problems but also promoting a proactive approach to system optimization. Observability allows your DevOps team to identify bottlenecks and unearth opportunities for improvement, even if the system is working as expected.

Many observability approaches incorporate AI and machine learning to rapidly analyze historical and real-time data in order to catch issues before they occur. By using these tools, DevOps teams can uncover new patterns in observability data that they might have missed, expanding their ability to catch potential issues and build more resilient systems.

WHY IS OBSERVABILITY IMPORTANT FOR YOUR BUSINESS?

The scale of data collection, the speed of cloud-native software tools and the adoption of containerized microservices have all greatly increased the productive capacity of modern systems. However, these changes have also brought a new level of complexity, and with it, new potential risk factors. IT and DevOps teams can no longer rely on simple monitoring tools, as the complexity of modern systems can make service downtime and mitigation lengthy and expensive.

The potential cost savings of being able to proactively detect issues means that observability could offer a significant return on investment. Observability can also help you maximize your service uptime, which can make your internal data processing and analytics more performant, leading to greater trust and insights throughout your organization. This is not only an asset to your technical teams but to your go-to-market strategy, sales and marketing efforts.

Observability is also useful for visualizing and understanding the breadth of your whole system, which is essential for data security and compliance. Following the movement of data through a highly complex set of distributed systems is difficult without having a means to track how these systems interact. An observability approach necessarily involves mapping the system out and updating it as you add on or deprecate tools and databases.

THE PILLARS OF OBSERVABILITY

When incorporating observability, you will draw from three main sources of data. Each of these sources is useful on its own, but it's the combination of all of them which allows you to achieve an actionable understanding of your system.

Metrics

Metrics are the most basic unit of observability, measuring different system performance characteristics. They typically have a simple structure, usually a numerical value along with the time of collection and the type of data it represents. Collecting and aggregating metrics like service uptime, CPU utilization and cache hit ratio can help provide an overview of system performance.

Logs

A more granular and detailed form of data than metrics, logs track processes that occur within a system or service. Logs include the type of process, for example, a database query, the time it ran and other contextual information. Rather than being triggered by a particular threshold, logs are a record of every single action a system performs.

Traces

You can use distributed tracing to understand how different services or applications interact with one another. It follows a single request or interaction through a system: for example, the process a transaction request goes through when a customer buys something on the front end. By tracking the way this request works through the purchase flow, collects the customer's information and payment data, processes a payment and pushes the order to the supplier or warehouse, your DevOps teams can use this information to uncover issues or replicate an error that a user reported.

Events

Sometimes included among the observability pillars (taken together, referred to as "MELT"), you use events to understand a particular interaction with a system in order to diagnose problems. Because they use a specific time stamp and other identifiers, this data lets you pinpoint a discrete event and explain what happened at that moment in time. For example, you can examine a login attempt or a manual database query as an event, giving your DevOps teams crucial context.

User Experience (UX)

Though not considered a pillar of observability, because observability includes system outputs, it can be useful to examine UX metrics in addition to the others listed above to get a fuller sense of how and why certain issues happen. UX can be a very useful way to set performance benchmarks, allowing you to see if an internal system change you made has impacted users positively or negatively. This can be important for complex systems and multicloud architectures, as users in different regions or on different platforms might have different experiences.

BENEFITS OF IMPLEMENTING OBSERVABILITY IN YOUR ORGANIZATION

As an all-inclusive approach to system assessment, observability can offer significant benefits to performance while increasing efficiency and saving resources. Here are some of the key ways an observability strategy could benefit you.

Faster incident detection and resolution

Because observability aggregates many different sources, it allows you to track critical metrics and events through your entire system. This makes it much more likely that your team will catch, isolate and resolve any issues that occur at any point in time, often before it can cascade into larger issues elsewhere. Observability is also flexible, allowing you to collect telemetry data from complex elements, including containerized and cloud-native tools and microservices.

Enhanced security posture

Having an end-to-end understanding of your system architecture can greatly improve your security efforts, helping your DevOps teams track data flows and reduce potential attack surfaces. It's also a useful tool for your security team, as they can design and conduct penetration tests to assess the security of new or critical parts of the system. This is especially helpful when clearing compliance thresholds or seeking out security certifications, as these might require you to control access to sensitive data and store identifiable data in a particular region, among other exigencies.

Better capacity planning

Having a holistic view of your system makes it much easier for you to uncover patterns in usage and capacity, allowing you to react to increases in demand for storage, compute and memory usage. For example, if you uncover a pattern of peak usage correlating with particular times of the day, you can use this to increase capacity and ensure that services are properly resourced.

HOW ENGINEERS AND DEVELOPERS BENEFIT FROM OBSERVABILITY

The benefits of observability can extend to your whole organization, but they are most evident in the technical work you do. The primary goal of observability systems is to make issue resolution faster and more efficient by helping your team uncover the root causes of a problem. Because of the multitude of data sources in most organizations, observability can greatly reduce the amount of investigation and contextualization you have to do when troubleshooting and can even automatically recommend and effect steps towards solving issues. All of this can reduce mean time to resolution (MTTR), which saves time and resources and shields end users from problems.

Another benefit of observability is that it allows you to head off any potential issues when your technical team changes a system, whether by adding a new microservice or functionality or deprecating one. Because observability provides a deep understanding of your systems, it fosters communication between teams around how they use different services and how a change could impact other teams or parts of the system. Observability becomes a preventative tool, per se, as it shows the dependencies and interactions throughout your operation, which can help your teams avoid changes which might negatively impact system health and uptime.

CHALLENGES FACED WHEN IMPLEMENTING OBSERVABILITY

Observability practices allow teams to effectively manage complex systems. However, an observability strategy requires significant investment and is not without its challenges. Here are some of the most common:

Data overload from extensive telemetry data

Gaining observability, at least in part, requires you to create new data collection processes to measure the performance of your old data collection and storage operations. This can become a scaling challenge, as the data you collect leads to more data than you can manage. Collecting extensive telemetry data can create even more data management issues, making it difficult to draw insights from that data.

Complexity of integrating multiple tools and platforms

Setting up observability instrumentation in a complex system can take a lot of time and resources, as you need to measure and log each individual element, integration and interaction. If you aren't able to develop processes to build these kinds of tools in a low-cost, repeatable way, you can overload your IT or DevOps teams with setup work.

Ensuring data security and compliance

A tool which continually monitors and aggregates a wide range of telemetry data and logs is necessarily going to create security risks. By making these kinds of system-wide assessments available, often as a single source of truth, this approach could become a single point of failure, and it can also represent a vector for data misuse or accidental leaks.

Managing costs associated with large-scale data collection

If your team is fully bought-in and adds observability tools to every extant and new service, any significant increase in service usage, user growth or other events can lead to a deluge of data hitting your observability tools. This can have a cascading effect on your budget as large amounts of data move through your system, leading to the creation of even more telemetry data.

HOW TO SUCCESSFULLY IMPLEMENT OBSERVABILITY IN YOUR ENVIRONMENT

The exact techniques and tools you use will depend on your particular architecture, but you can follow these general steps to implement a robust and flexible observability system within your organization:

Assess current monitoring capabilities and gaps

Every modern organization has some form of basic monitoring and may even have invested in observability components like metric collection. You need to map out your system from on-premises and cloud storage to edge services, looking at the different ways those services interact to understand exactly what kind of data you will need to collect and how often.

Choose appropriate instrumentation tools and platforms

As you build an end-to-end understanding of your organization, you will need to create benchmarks and estimates of how much data you'll need to collect, store and process to achieve observability. You should also choose solutions which offer the most flexibility, particularly open-source tools which are vendor agnostic and can scale as your data needs grow.

Collect comprehensive telemetry data

Once you've selected tools appropriate to each use case, your team can implement these changes, and you can begin to collect telemetry data. Embracing solutions that are repeatable and can automatically adapt to different use cases is important for saving your team's time and resources.

Set up dashboards and alerts for real-time visibility

Once you have established the way data flows in your operation, you can begin to visualize insights from that data flow. A dashboard offers a centralized way to track issues, trace processes and uncover architectural risks and inefficiencies. You should use a solution that you can change and augment as you identify new risk factors and incorporate new components.

Train teams on interpreting data and responding to issues

Achieving team buy-in is critical, and you should resource your DevOps teams, security teams and other stakeholders to fully understand your observability strategy. As you ramp up data collection, you should train them and build out an incident response process that utilizes the breadth of data and analysis tools observability offers you.

Continuously improve based on feedback and evolving needs

Observability requires continuous improvement as your system changes, and you will need to work across teams to identify the best way to implement data collection, logging and other observability components.

THREE FEATURES TO LOOK FOR IN AN OBSERVABILITY TOOL

Choosing the best observability tool will require you to have a deep understanding of both the available solutions and the unique needs of your system. Here are three key features to look for:

Advanced analytics and AI capabilities

Although observability offers more flexibility than legacy monitoring processes, the pace and complexity of modern software development can make observability maintenance labor-intensive and demanding. Observability tools that can use analytics insights and AI to adapt to new issues and perform system maintenance can allow your team to dedicate more time to innovation and efficiency.

Scalability to handle growing data volumes

As your central issue detection and solution system, it's critical that your observability tools never lead to data overload. Misallocating computational and storage resources, whether providing too much or too little, can mean missing critical issues, lengthening MTTR and increasing costs at an unsustainable rate.

Strong security features and compliance support

Tools that utilize automation and AI to reduce manual data correlation and can automatically handle common reliability issues can give your security team more time to solve more complex issues quickly. Other features, such as sensitive data redaction and robust permission controls around telemetry data access can help you limit the risk of a data leak.

CONCLUSION

Harnessing a broad range of software development and data storage solutions means you must embrace some level of complexity. The diversity of different cloud-based solutions, as well as containerization, serverless architecture, multicloud models and other tools, has made it impossible to stretch legacy support and issue detection systems to cover every part of your system.

Observability is a better approach that avoids overreliance on one metric, indicator or event, instead giving you the ability to see the entire system as it functions and helping you zero in on root causes quickly. By embracing an efficient and performant observability solution like Snowflake Trail, you can enjoy the benefits of complexity without sacrificing performance, MTTR or service uptime.

OBSERVABILITY FAQS

› **What are observability solutions?**

› **What is an observable system?**

CUSTOMERS USING SNOWFLAKE

simon data logo

WHOOP Improves AI/ML Financial Forecasting While Enhancing Members’ Experiences

With Snowflake and Apache Iceberg, WHOOP teams have centralized access to data while reducing complexity, lowering costs and improving critical processes.

Read the story ›

penske logo

Pfizer Accelerates Insights and Lowers TCO by 57% While Processing Data 4x Faster with Snowpark

By migrating to Snowflake, this pharma giant has unified business units with greater access to insights and seamless data sharing — all while reducing total cost of ownership (TCO).

Read the story ›

OBSERVABILITY RESOURCES

● PRODUCT

Snowflake for Applications and Collaboration

Explore the solution ›

● REPORT

Building a Startup in the Age of AI

Explore the feature ›

● EBOOK

The Essential Guide to Data Sharing

Get the ebook ›

● EBOOK

Design Patterns for Building Multi-Tenant Applications on Snowflake

Get the ebook ›

● EBOOK

50 Essential Apps, AI and Data Products on Snowflake Marketplace

Get the ebook ›

RELATED CONTENT

What Is Big Data? Characteristics, Benefits and Examples

Discover the benefits, challenges and use cases of big data and its substantial impact on innovation.

Scala vs. Python: Key Differences & Use Cases

Compare Scala vs. Python for data engineering. Explore their use cases, performance, and how both languages work together in modern workflows.

What Is Predictive Analytics? Models, Tools & Use Cases

Discover predictive analytics models, tools and use cases. Learn how businesses use predictive analytics to forecast trends and drive decisions.

Data Analytics in Manufacturing: Use Cases & Benefits

Learn how data analytics in manufacturing helps improve forecasting, optimize supply chains, and reduce downtime. Explore manufacturing analytics software.

What is Customer Segmentation? Analysis, Models & Examples

Discover what customer segmentation is, its strategies, types, customer segmentation examples and how it boosts business growth and targeted marketing.

AI in Investment Banking: Top Use Cases & Benefits

Explore how AI in investment banking is driving automation, portfolio optimization, and digital transformation with real-world use cases.

What is a Regression Model? Definition, Types & Examples

Learn about regression models in ML with real-world use cases. Understand regression definition, types & examples of regression analysis.

What Are OLAP Cubes? OLAP Meaning and Use Cases

What are OLAP cubes? Learn OLAP meaning, use cases, and how data cubes help power fast, multidimensional analysis in business intelligence.

What Is a Transactional Database? Properties & Use Cases

Transactional databases keep the digital world running smoothly behind the scenes. From online banking to retail checkouts and healthcare records, these systems are built for speed, accuracy, and reliability—ensuring every transaction is processed securely and without error, even under heavy load.

Product

- Platform
- Snowflake Intelligence
- Data Engineering
- Analytics
- AI
- Applications & Collaboration
- Pricing

Support

- Support
- Priority Support
- Status

Industries

- Advertising, Media & Entertainment
- Financial Services
- Healthcare & Life Sciences
- Manufacturing
- Public Sector
- Retail & Consumer Goods
- Technology

Company

- About Snowflake
- Leadership & Board
- Careers
- Investor Relations
- Trust Center
- Brand Guidelines
- Contact
- Newsroom
- Environmental, Social & Governance
- Snowflake Ventures
- End Data Disparity
- Snowflake Summit 26

Learn

- Resource Library
- Live Demos
- Fundamentals
- Training
- Certifications
- Snowflake University
- Developer Guides
- Documentation

Subscribe to our monthly newsletter

Stay up to date on Snowflake's latest products, expert insights and resources—right in your inbox!

Business Email Address

Company Name

Country

☐ Add me to the list to receive dedicated product updates and general availability emails.

By submitting this form, I understand Snowflake will process my personal information in accordance with its Privacy Notice. I may unsubscribe through unsubscribe links at any time.

SUBSCRIBE NOW

© 2026 Snowflake Inc. All Rights Reserved