LogicMonitor

EBOOK

The Ultimate Guide to Observability

Tips, tricks, and key concepts for IT organizations looking to start their journey to an observable future.

Table of Contents

Chapter 1: What is observability

Not so long ago, IT professionals had a clearly defined focus. They were working with a limited, monolithic set of applications that resided in one place—the data center—and could be cared for like pets. When one got sick, IT nursed them back to health. Today, the pieces are smaller, widely dispersed, and are more like cattle than pets, because if something goes wrong, IT removes it from the herd. IT is managing increasingly hybrid environments, with a mix of resources in the cloud, and others running on-premises in physical data centers. In this multi-cloud world, network environments are becoming more complicated, complex, and opaque.

By 2022, more than 50% of enterprise data will be created and processed outside the data center or cloud, up from less than 10% in 2019.[1]

The speed of innovation is accelerating, driving further infrastructure changes. As the pressure grows on organizations to innovate faster and get products and service offerings to market more rapidly, DevOps is becoming increasingly fundamental to business outcomes. DevOps teams are embracing the cloud, and the demand for cloud-native technology is escalating as serverless and container platforms grow.

Forrester predicts 25% of developers will use serverless and nearly 30% will use containers regularly by the end of 2021, creating a spike in global demand for both multi-cloud container development platforms and public-cloud container/serverless services.[2]

This fundamental shift from "pets to cattle" makes it exponentially more challenging to proactively identify problems and repair them before they affect users. Today's environments are not only complex, but a constantly shape-shifting polyglot. Few IT shops have the luxury of existing solely in one world; most are forced to operate in both. That's not easy, because the tools designed for one environment are often poorly-suited for use in the other.

As network environments and business imperatives are changing, IT is becoming a more strategic, creative force. ITOps professionals are no longer focusing on their old mission of simply keeping the lights on, but on providing the resources that line of business needs to drive growth and power high-profile initiatives. In many cases, it's no longer good enough to merely monitor system availability; business owners demand near-perfect quality of service.is needed to implement observability across your organization.

[1] https://www.gartner.com/smarterwithgartner/gartner-top-10-trends-impacting-infrastructure-operations-for-2020/

[2] https://go.forrester.com/blogs/predictions-2021-cloud-computing-powers-pandemic-recovery/

The combination of new responsibilities and decentralized environments are bringing new pressures to bear on IT professionals at every level. Whether they are working on the front lines as practitioners or setting strategic priorities as executives, IT professionals need improved ability to see and make sense of all the pieces within their apps and infrastructures, so they can make better, faster decisions. They need to move beyond mere monitoring to embrace "Observability."

Observability is a somewhat new term in the IT monitoring space. Its academic definition is the ability to measure the internal states of a system by examining its outputs. A system is considered "observable" if the current state can be estimated using only information from outputs, namely sensor data. In the real world, observability is an outgrowth of IT monitoring. It means the surface area of things to monitor continues to grow, and it takes metrics, traces, and logs to answer the questions of:

1. What's going on? Do I have a problem?
2. Where is the problem centered?
3. Why is the problem happening? (root cause analysis)

Observability covers all layers of an app or service, beginning with the infrastructure and network and building up to the application layer. On top of that, observability also measures business outcomes, often as closely as do any other underlying inputs such as disk consumption.

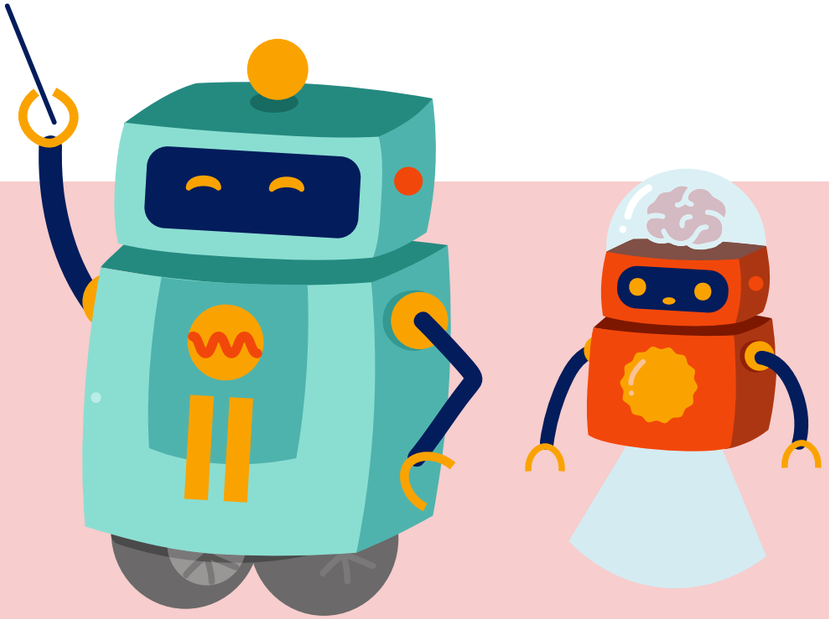
As systems become observable, IT professionals gain insights into "unknown unknowns," allowing them to better sense the true quality of service — and proactively identify and repair issues when they do arise.

Let's take a closer look at what observability means for ITOps, DevOps, and MSP professionals. We'll show you how it can benefit your business and set the stage for new levels of performance, availability, and service quality. And we'll walk through some of the essentials of what's needed to implement observability across your organization.

What is o11y anyway?

Did you know that "o11y" is merely shorthand for "**observability**"? Officially known as a "numeronym," it combines the **11 letters** between o----------y and is simpler to type. We're a lazy bunch in our industry. Plus, we like terminology that no one else can understand! Many have even taken to pronouncing the abbreviation itself: "**olly**" by turning the 1's into L's.

The concept of IT contractions derives from Andreessen Horowitz, which folks started abbreviating as **a16z** because it was long and difficult to type. Then the nomenclature expanded to other hard-to-spell/type words in our domain, like **i18n** (internationalization), **l10n** (localization), or **k8s** (Kubernetes).



Unifying tools and teams to achieve observability

Monitoring has never been simple, but there was a time when it was certainly simpler. ITOps professionals had a device they could collect data from—and knew the metrics they needed to monitor. If something went wrong, they could find the root cause. They knew where to look and what types of data would lead them to the source of an issue. But with more things to monitor, more updates, more data, more movement, more everything; monitoring in general needed to grow with it. Methods that served us well in former times no longer scale to the needs of modern environments.

Observability is the answer to simplifying the mounting complexity of IT. It goes beyond monitoring, which creates visibility into environments and is a foundational piece of IT operations. But monitoring doesn't deliver total observability. Monitoring is anchored primarily on metrics (and alerting when they violate preset thresholds). And solving problems based on metrics alone is incredibly challenging. Think of it: a metric showing a high CPU or full disk doesn't show root cause, and thus is not actionable. Capabilities such as logging, contextual tracing, and machine learning need to be added to create an observable platform. In short, monitoring is one key function of observability, but observability has additional components that ensure teams can move from reactive problem solving to proactive operations—no matter how complex the pieces become.

Observability in IT refers to a system's ability to diagnose what's happening inside a system or process by observing the external visible information available. This is done by combining monitoring, log analysis, and machine learning to create an environment that can easily detect issues, proactively identify anomalies, and scale with your growth.

Monitoring

The first tenet of an observable system is monitoring itself. Being observable requires as close to total visibility into an IT environment as possible. To correlate data points, devices, and functions, IT needs to be able to see everything.

Monitoring is founded upon principles of time-series data comprised of metrics. When metrics run above or below a set threshold, an alert is generated and sent to an operator for review. In modern systems, thresholds are often dynamic and adjust automatically based on volume, time period, or other conditions. What is red at one point in time doesn't necessarily mean it is red at another time.

Log analysis

Logs provide context and are required to find issues and recognize trends in observable platforms. Logs are often thought of as answering the "why" part of the triage process. Having records of every event across a vast array of data is necessary to help know what's happened previously and quickly find errors when something goes wrong. Thus, in observable systems, the analyzed quantity of data is often exponentially larger than in the monitoring of previous generations.

Machine learning

To automate the correlation between metric data and log data, machine learning can be used to detect, analyze, and provide insight. Thousands of logs, metrics, or traces can be created every second, across thousands of devices. Separating signal from noise is impossible for humans alone. With machine learning, this data can be analyzed to find anomalies, errors, and issues quickly, sometimes even before a problem arises.

A holistic approach of teams and tools

Observability requires a sophisticated toolset and holistic approach to monitor, analyze, and trace events. The chapters ahead will provide a detailed look at implementation, but here's a simplified overview of the basics of implementing observability:

Start with centralized monitoring

Organizations should start by achieving as much visibility into their environment as possible. Monitoring is a foundational piece of observability, and a good monitoring platform should provide insight across all environments; whether they are on-premise, in the cloud, or hybrid. Monitoring should also cover all layers of an app or service, starting with the network and infrastructure and moving up to the application layer and even the business results that those layers create. In some cases, external metrics can also help enrich the story. Data such as weather, traffic, and social sentiment can be part of a strong observability strategy if those metrics can help correlate changes in user or system performance.

The more organizations monitor and the broader the scope, the more data they have, and the more they will be able to streamline troubleshooting and reduce mean time to resolution (MTTR). Monitoring answers the "what and where" of observability. Metrics tell teams that they have a problem, and traces often tell them where issues occur.

That said, it's important to focus on centralized monitoring so that all devices and environments (whether they are switches, routers, serverless, cloud instances, etc.) can be quickly diagnosed within the same pane of glass. Having one monitoring platform to see AWS deployments and another to see network devices will slow down the troubleshooting process and won't provide a full picture of where performance issues are happening.

Analyze logs

Analyzing logs manually is cumbersome, and logging every single event quickly creates millions of lines to parse through. In fact, this problem is just going to get bigger. According to IDC, the "digital universe" is growing 40% every year, but most of us only analyze about 1% of that data. We have all of this data, but we lack the mechanisms to make use of it. That's why it's crucial to have a tool that goes beyond rule-based policies for reactive discovery of logs and rather analyzes all logs in real-time to surface key events.

Marrying monitoring and log analysis can help to filter out and sort the worthwhile data (critical log events, anomalies, etc) from the junk; saving a ton of time and eliminating manual processes. With real-time log analysis, organizations can answer why and how something went wrong in an IT environment—pushing them one step further towards full observability.

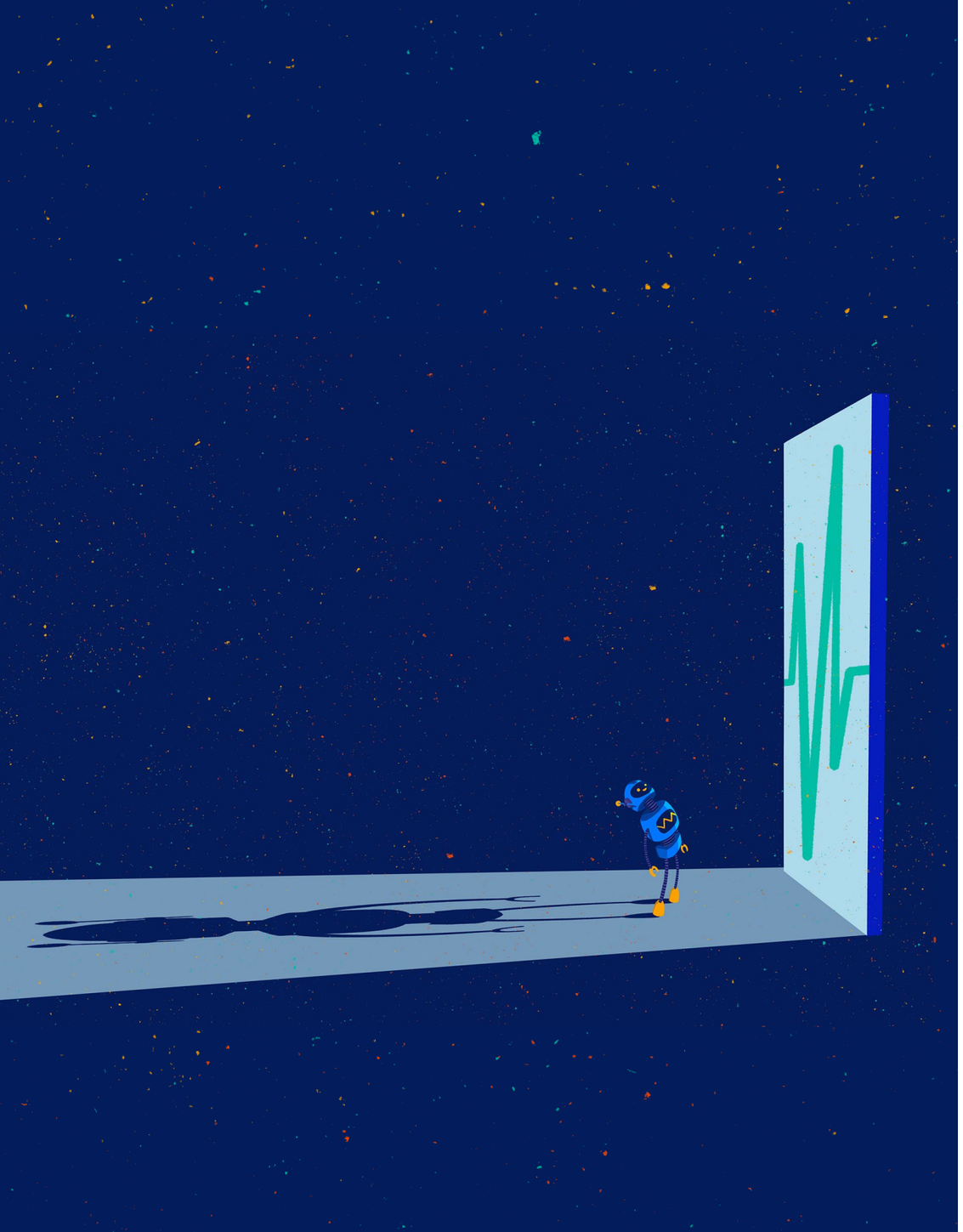
Implement algorithms

Finally, organizations must answer the question of what to do with all of this information. The whole point of collecting and correlating all that data is to create a manifest action: do something. With log analysis and centralized monitoring, they now know where an issue occurs, how it happened, and why it caused a particular chain of events. By adding in machine learning and taking an algorithmic approach to IT operations, they can automate the exponential growth of data into actionable insights that allow them to do more in less time. These processes can help to streamline IT operations by setting dynamic thresholds, identifying anomalies, and finding the root cause of an issue.

Over time, machine learning and algorithmic IT Operation systems (such as AIOps) can proactively identify issues by understanding what's normal for an environment and surfacing an alarm before that issue causes downtime. Therefore, avoiding service-impacting incidents is a mark of observability working well. Once an ITOps professional knows what to do with all of your metrics, traces, and logs, they can hope to achieve full observability. High-performing organizations don't let the same problems happen repeatedly. They isolate the cause and automate responses. That's a tenet of how Site Reliability Engineers (SREs) use principles of observability to mature the organization.

Observability requires a sophisticated and holistic approach to using a collection of tools to monitor, analyze, and trace events. Once organizations achieve observability, their team will be able to keep crucial systems and business applications up and available, enabling the business to innovate and advance forward.

©LogicMonitor 2021



Observability vs. Monitoring

Observability and monitoring are sometimes used interchangeably, but they aren't the same. The ability to achieve observability may somewhat rely on effective monitoring tools and platforms. It could be said that effective monitoring tools augment the observability of a system.

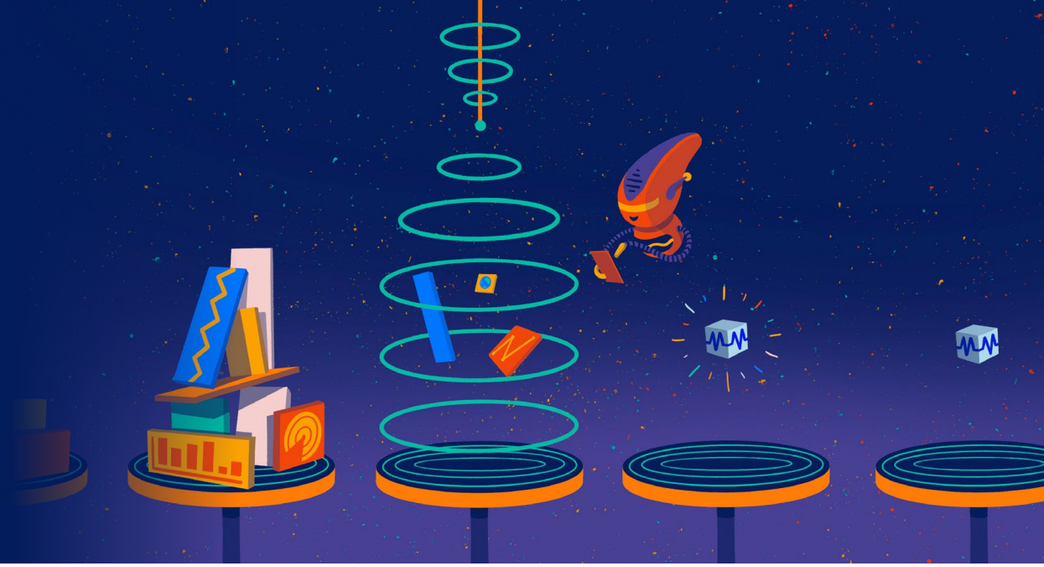
Monitoring is an activity someone does. The action monitors the effectiveness or performance of a system, either manually or by using various forms of automation. Monitoring is anchored primarily on "known knowns" such as a CPU being 100% or a disk being full. However, monitoring is also built on "known unknowns" such as "if the response time is much slower than normal for this time of day." These examples both begin with known metrics to monitor. Tools for monitoring collate and analyze data from a system or systems, provide insights, and suggest actions or adjustments where necessary. Some monitoring tools can provide basic but crucial information like alerting a system administrator when a system goes down, or conversely when it goes live again. Other monitoring tools might measure latency, traffic, or other aspects of a system that can affect the overall user experience. More advanced tools may link to dozens, hundreds, even thousands of data streams providing broad-ranging data and analysis for complex systems by dynamically comparing what is normally expected performance.

Observability isn't a form of monitoring, but could be described as something like "monitoring plus." As we move into monitoring "unknown unknowns," observability develops not only a description of the system, but also correlates that two or more seemingly disparate pieces of data (whether metrics, traces, and logs) are correlated in a meaningful way. It also determines whether the pattern the system observed indicates some sort of impact on outcomes. Thus, observability helps IT know where to look while weeding through volumes of data that it could never digest on its own.

The observability of a system determines how well an observer can assess the system's internal state simply by monitoring the outputs. Observability is when a developer can use the outputs of a system — for example, those provided by APM — to accurately infer the holistic performance of the system.

Thus, monitoring is not a synonym for observability or vice versa. It's possible to monitor a system yet still not achieve observability—especially if the monitoring is ineffective or the outputs of the system don't provide the right data. The right monitoring tool(s) or platform(s) help a system achieve observability.

Chapter 2: Pillars of observability: metrics, logs, and traces



A common way to discuss observability is to break it down into three types of telemetry: metrics, logs, and traces. These three critical data points are often referred to as the three pillars of observability. It's important to remember that although these pillars are key to achieving observability, they are only the telemetry—and not the result.

Metrics

Metrics are various values monitored over a period of time. Metrics may be key performance indicators (KPIs), CPU capacity, memory, or any other measurement of the health and performance of a system. Understanding fluctuations in performance over time helps IT teams better understand the user experience, which in turn helps them improve it.

Logs

Like the captain's log on a ship, logs in the technology and development world provide a written record of events within a system. Logs are time-stamped and may come in a variety of formats, including binary and plain text. There are also structured logs that combine text and metadata and are often easier to query. A log can be the quickest way to access what's gone wrong within a system.

Traces

A trace is a way to record a user request right along its journey, from the user interface throughout the system, and back to the user when they receive confirmation that their request has been processed. Every relevant operation performed upon the request is recorded as part of the trace. In a complex system, a single request may go through dozens of microservices. Each of these separate operations, or spans, contains crucial data that becomes a part of the trace. Traces are critical for identifying bottlenecks in systems or seeing where a process broke.

Correlating metrics

Even using this telemetry, observability isn't guaranteed. But obtaining detailed metrics, traces, and logs is a great way to approach observability. There is some crossover between these different types of telemetry, especially in the data they provide.

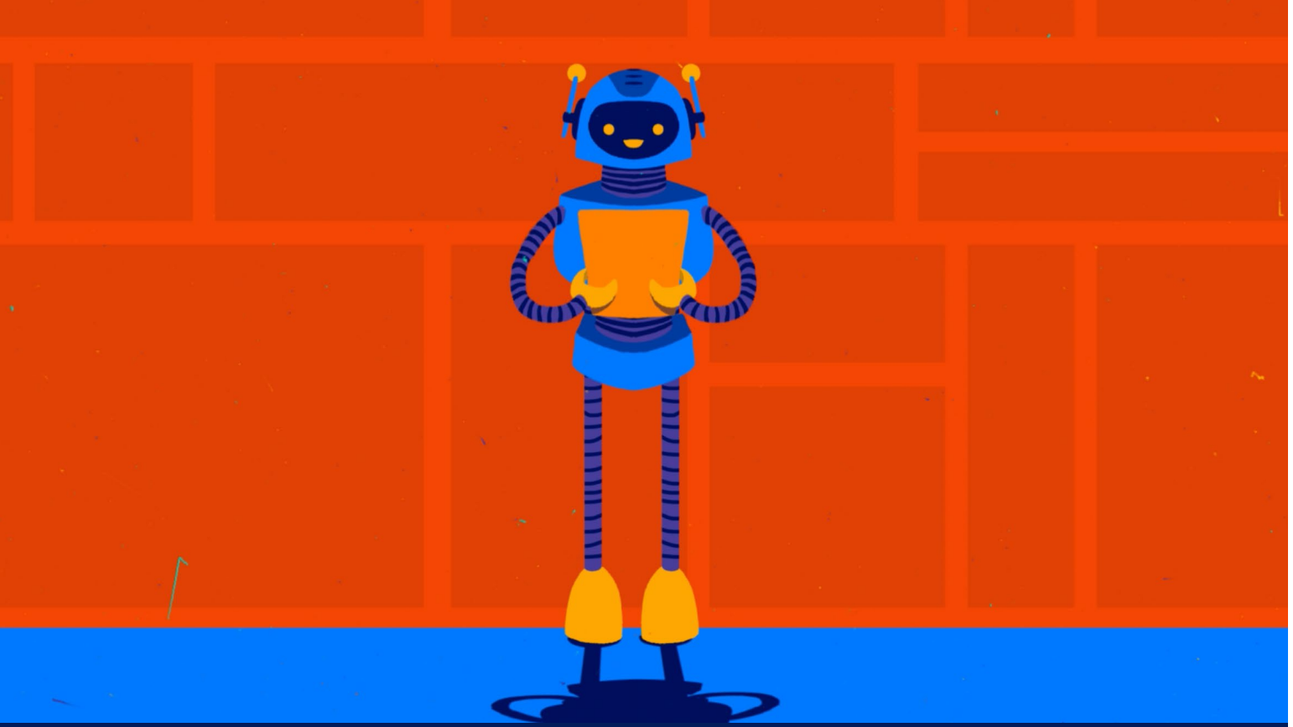
For example, metrics tracking latency may provide information that is similar to a trace-set focusing on the user. Adding the metrics to the trace set might provide additional insights into where latency occurs in the system. That's why it's important to view observability as a holistic solution — a view of the system as a whole but made up of various types of telemetry.

Events are another type of telemetry that can be used to help achieve observability. Often, metrics, traces, and logs are used to provide a cohesive view of system events, so they can be considered as more detailed telemetry of the original data provided by various events within a system.

Dependencies or dependency maps give the viewer an understanding of how each component of a system relies on other components. This helps with resource management, as ITOps can clearly understand which applications and environments are using what IT resources within a system.

Regardless of which exact types of telemetry are used, observability can only be achieved by combining and correlating various forms of data to create this "big picture" view. Any single one of the pillars of observability on its own provides very little value in terms of visibility and maintenance of a system.

©LogicMonitor 2021



Chapter 3: What are the benefits of observability

We've discussed some of the challenges that ITOps, DevOps, and MSP professionals face as they manage and update increasingly diverse, complex environments and systems. Observability provides a holistic way for people to address these challenges by gaining a view of the system as a whole, which can drive changes in behavior. For example, instead of just monitoring (and alerting) in the event of high CPU activity or a full disk, observability can point us toward Digital Experience Monitoring (DEM), watching users and triggering issuesbased on their happiness level. If the CPU is running high but users are happy, then there's no problem. In this situation, observability is using a triangulation of data from different parts in the value stack to determine what to do.

Observability has tremendous potential to drive measurable benefits and/or better-informed decisions for developers, IT professionals, and their entire businesses. In this chapter, we'll explore some of the positive outcomes that observability delivers across a variety of environments and use cases.

Building a more innovative, future-ready organization

Observability provides insights across multiple systems and environments, so its benefits are capable of impacting a wide range of areas.

One of its most compelling outcomes is cost control. The majority of decisions in any business are driven by concerns about cost or profit. The more costly an app or system is to develop, operate and update, the more that cost must be potentially passed on to the consumer. That can reduce the apparent value of systems, so anything that keeps costs down and increases profits is welcome.

For example, observability helps organizations reduce ITOps and DevOps costs by providing a better, more intuitive understanding of the internal states of systems. Effective data streams paired with data analytics and the right monitoring platforms mean less manual observation is required. At the same time, these insights enable updates to happen faster and with more confidence. This helps reduce the number of employee hours required to keep a system at peak performance, while faster, more effective updates increase the value of the system as a whole.

Turbo-charging business agility

Improved understanding enables organizations to accelerate decisions, processes, and innovation. Going fast, with confidence—and a perceived safety net that observability provides—empowers companies to take calculated risks so they can outmaneuver the competition. An organization can fearlessly rush a new capability to market because they know they'll be able to see and respond immediately if something goes wrong with it.

Companies can use that same telemetry data observability practices to find and fix problems in pre-production and avoid regular production impacts. Taken all together, observability makes organizations more future-ready, because they can respond more aggressively to changing market conditions.

Acquiring the big picture

For IT and development teams, observability has a unique ability to provide a shared view of an entire system. The view can encompass its health, performance, architecture, and services and processes.

With observability, developers, operators, and other team members can take advantage of a common context that spans their organizations and roles. They can be confident that they are interacting with the same data and insights into services, alerts and incidents, and other critical views into their systems.

Because observability tools are continually capturing data at particular points in time, they can help organizations develop a historical record of how their environments and processes have evolved and changed. IT and DevOps professionals can easily go back and review past records of system health, architectural changes, or other details about their systems to gain insight from the past—and make better decisions going forward.

By providing a more complete record of what has happened, observability can support AI and ML capabilities to help organizations better predict what might happen next. More importantly, it lets organizations discover small changes that might be otherwise invisible—much like the old analogy of a frog slowly boiling, instead of immediately. For example, if applications are steadily slowing down by only a small amount each month, the ability to see that trend, and then tie it to underlying components can help organizations tackle small issues before they become big headaches.

Improving perspectives for ITOps

For professionals working on the front lines of IT Operations such as IT analysts, system engineers, and infrastructure architects, effective execution is top of mind. These individuals are routinely building dashboards, managing alerts, and responding to issues using monitoring solutions, but are also striving to work more strategically. They would no doubt prefer to spend less time fighting fires, and more time focusing on larger-scale projects and innovating for customers.

For ITOps professionals, observability can provide a foundation for AIOps (Artificial Intelligence for IT operations) and monitoring. In recent years we have seen the emergence of observability platforms, or the ability to monitor metric, application, and log data in a single platform approach. Recently observability has shown importance, as it's not only about monitoring these data sets but ensuring that they are correlated and in context of each other. AIOps capabilities are then applied to automatically filter out the noise from the streams of data, to enable early warning of issues before widespread business impact, and to proactively prevent failures such as outages.

"Unified Observability is not only about monitoring data sets but ensuring that they are correlated and in context of each other."

For ITOps professionals who must find a needle in an entire field full of haystacks, observability not only points to the most likely haystack but provides clear direction about the part of the haystack where the needle is most likely to be found.

There has long been an unspoken concern among IT practitioners that as the tools get smarter and more outcome-oriented, there's an ever-increasing likelihood that the person in the equation will be replaced. Some solutions vendors have even reported anecdotes of customers actively sabotaging sales cycles because they are worried about being replaced.

However, in reality, observability is not about diminishing the role of people, but all about up-leveling work from "less-fun stuff" to "more-fun stuff." Observability tools can help practitioners to move towards outcome-based work that is more rewarding for the people actually performing the tasks. For example, instead of scouring log files by hand looking for specific patterns, an IT administrator could have a platform do the job for them and instead focus on how anomalies correlate to other data. Smart practitioners have seen the writing on the wall, and are already aligning themselves with SRE practices to stay relevant as manual, time-consuming tasks are increasingly automated.

Observability can also support improved post-mortem investigations following incidents. By providing a historical record, observability can enable ITOps to review specific details about system behavior at the time of an incident, instead of depending on individual anecdotes and recollections that may not be accurate.

Empowering DevOps and engineers for fearless deployments

Observability is about enabling developers and engineers to do their jobs better. When developers and operations teams don't have to spend hours diving deep into the internal workings of a system, they have more productive time available. This is time that can be spent creating better ways for users to engage with the metrics that matter and improving the overall user experience.

©LogicMonitor 2021

This has significant financial implications, as the better a system works, the more desirable it is to consumers. Even free-of-charge software and apps can help build a company's brand reputation and revenue through the usability of their systems working seamlessly together and providing more time-to-value to development teams.

More observable systems are easier to understand, control, and optimize compared to those that lack observability. Observability brings end-to-end visibility for cloud-native applications to developers, engineers, and operations to enable fearless deployments, and deliver a world-class user experience. Furthermore, as systems change through new software deployments or scaling the organization up or down, observability can provide an improved perspective. By establishing a historical baseline, observability can help DevOps pros better understand when the system starts to drift from its ideal state. Improving perspective can also impact dev team morale because they can spend more time building features that matter. Every developer wants an assurance that their work matters and observability practices can help developers find and fix problems fast, yielding more time for "real work" tasks that they will find rewarding.

Deep visibility of the internal workings of a system, combined with rich data and event analysis helps DevOps deal with what's happening right now—and helps them project future events and plan effectively for the future. This can be via understanding potential peak times and capacity fluctuations, allowing for the effective reallocation of resources. This can also alert DevOps to quieter times when testing of new services might be more appropriate. Plus, having an observable system means those tests become more effective, and less likely to cause a system crash or downtime.

The more experienced development teams are at applying observability practices, the more apt they are to "shift left" their work. They will test earlier and find (and fix) problems far sooner in the development process because the tooling and data are designed for that outcome. Observability lets developers avoid the need to wait until they are in production, and ask their operations team to scour logs when issues occur. When combined with advanced automation and strong CI/CD pipeline processes, observability can enable organizations to move beyond traditional development approaches to shift left.

"Observability promotes stability, something consumers expect more and more from the apps and software they use daily."

Maximizing service availability and the MSP experience

MSP professionals are relentlessly focused on repeatability and managing their organization's margin. Their business model is based on delivering services better than their customers could themselves. MSPs have the advantage of tools and repeatable processes, together with standardization, that supports a lower total cost of ownership (TCO). Although observability isn't unique to MSPs, its practices can help them support the standardization and repeatable processes they require, and which have become increasingly difficult to maintain as their environments become more complex and dynamic. For example, an MSP can't simply rebuild a monolithic server or application over and over. Their service offerings are based on a combination of multiple moving parts. Observability practices are an important principle to enable them to maintain margins on those applications.

As with every organization, availability is also key for MSPs. Higher-level individuals like service delivery managers and directors are more focused on overseeing the customer experience, onboarding, and managing multiple issues and escalations. Whether they are administrators, engineers, managers, or executives, MSP professionals at every level are short of time, with a lot to manage—and time is money.

Whether a business's systems are completely internal or used to provide services to consumers, downtime can be devastating. It's great to have contingency plans in place for when IT crises occur, but it's even better to have an observability solution that allows systems to stay online for the majority of the time. Having a detailed yet holistic understanding of the state of a system allows changes and updates to be made promptly to deal with issues that may otherwise cause the system to go down entirely. Observability promotes stability, something consumers expect more and more from the apps and software they use daily.

Chapter 4:
Why OpenTelemetry matters for observability

Tech innovators are increasingly focused on delivering more efficient, relevant digital experiences. Expectations are rising, and today's consumers and end-users continue to expect more out of every interaction, regardless of their locations and devices they are using to engage. As technology continues to advance, companies that cannot provide these kinds of personalized interactions for their customers will find themselves falling behind the competition. However, the cost of instrumenting (and re-instrumenting) applications has created risks that IT leaders have long sought to avoid. The cost of ownership associated with maintaining or changing agents for data collection has long figured into whether an IT organization can adopt new technology aggressively. OpenTelemetry provides a standardized method for data collection that helps organizations minimize the risk of making a bad (and expensive) choice. Companies are also beginning to use OpenTelemetry to collect data on customers' application stacks that can be applied to help elevate their overall experience. When companies make good use of that information, they not only help the customers who want to work with them but also develop more knowledge of their potential target market requirements.

OpenTelemetry and observability allow companies to optimize the customer experience by discovering slow processes or identifying areas of applications that might be more error-prone. Put simply, OpenTelemetry provides companies with a consistent mechanism for data collection and formatting. Instead of spending time collecting critical telemetry through multiple diverse applications, companies can focus on more important things like delivering new features and faster container deployments.

In this chapter, we'll take a closer look at OpenTelemetry, how it works, and its relationship to observability. We'll explore why companies find so much value in it and discuss how they can best use it to their advantage.

What is OpenTelemetry?

Technology leaders worldwide understand the value of digital transformation. However, digital transformation is not possible without the right tools and techniques. To move forward, it is essential that they fully understand the needs of their customers and their expectations for online and digital experiences.

The new, vendor-neutral standard for the collection of telemetry data, OpenTelemetry can be used for the collection of this data across applications, along with the services and supporting infrastructures for those applications. OpenTelemetry is part of the Cloud Native Computing Foundation (CNCF), and has enthusiastic support from many enterprise software vendors, ensuring that the project will be applicable to many types of technology and projects. Often referred to as "Otel," OpenTelemetry builds the equivalent of a "lingua franca" for those wishing to understand the performance of applications and services and the underlying components that support them—no matter how or where they are built, or by whom.

OpenTelemetry works with cloud technologies, orchestration engines, and containers to allow for faster innovation in the digital sphere. It also offers flexibility by allowing enterprises to standardize their way of collecting data with less vendor lock-in and more interoperability. This will be the trend moving forward.

OpenTelemetry architecture essentials

OpenTelemetry is focused on the collection of telemetry data, not the analysis. While Otel is capable of doing limited analysis-like activities within its collectors, its value lies in the standardization and ubiquitous collection of telemetry data.

The architecture involved in OpenTelemetry consists of several components to provide maximum value to organizations that use it. Each performs a specific role in enabling organizations to support data collection and usage for metrics.

API — APIs are computer language-specific, and provide the overall basics for adding OpenTelemetry. They make it easier to add additional capabilities to an application by having a framework that is available for attachment.

SDK — Another language-specific component of the architecture is the SDK, which provides a bridge between the exporter and the APIs. Transaction sampling and request filtering are both handled more easily due to the additional configuration of SDK.

Collector — Although not a requirement, a collector makes adding OpenTelemetry architecture much faster and easier. This component provides flexibility when sending and receiving application telemetry on the backend. Collectors solve problems with data egress and aggregation, sampling, and other routing/manipulation tasks. They can be deployed either as agents or as a standalone model.

Exporter — An exporter lets companies determine which backend(s) they are sending telemetry to. The backend configuration is not coupled to instrumentation, allowing for several different or simultaneous choices. Switching backends is easy, and there is no need to re-instrument code to do so. For example, an instrumented application that sends its telemetry data to a collector could easily be updated to forward that data to both an open-source backend such as Zipkin or Jaeger, as well as a commercial APM backend without requiring any changes to the instrumentation or collection layer, providing tremendous flexibility.

Benefits of OpenTelemetry

The true value of OpenTelemetry is in the benefits it provides to companies that use it for tracing and metrics. Ultimately, the company's end customers will reap the full benefits, even if they are unaware of the mechanisms that are delivering them. They simply enjoy feeling heard and seen and having a digital experience that encourages them to feel valued and appreciated by the company.

While the collection of application telemetry is far from new, OpenTelemetry provides companies with a consistent mechanism for collection and format. That is something that other telemetry providers do not offer and can make it very difficult to switch backends, processes, systems, and anything else that may be necessary as a company grows and develops.

When trying to understand the health of applications being used by a company, operations personnel, developers, and troubleshooters often find that the lack of consistency becomes incredibly frustrating. Applications also fail to work well with others, and upgrading or transferring from one application to another may result in a loss of data and metrics or the inability to trace them correctly.

OpenTelemetry can help make these issues a thing of the past. The standard created for adding a higher level of observability to cloud-based applications is a vital part of the benefits for most companies. Additionally, organizations will no longer have to spend any of their time attempting to develop mechanisms to collect critical telemetry by cobbling together separate applications. Instead, they can focus on the delivery of new features delivering a superior customer experience.

OpenTelemetry enables teams to go through the process of instrumenting their applications to generate high fidelity logs, metrics, and traces in an open format, so that they can utilize different tools to analyze it as needed. In the past, this level of instrumentation was often impractical, because it frequently locked organizations in with a certain vendor, minimizing flexibility and requiring new instrumentation setups in the event of a vendor switch. Broader adoption of OpenTelemetry also means easier and faster container deployments for companies, because there is no need to build an orchestration platform at the enterprise level first.

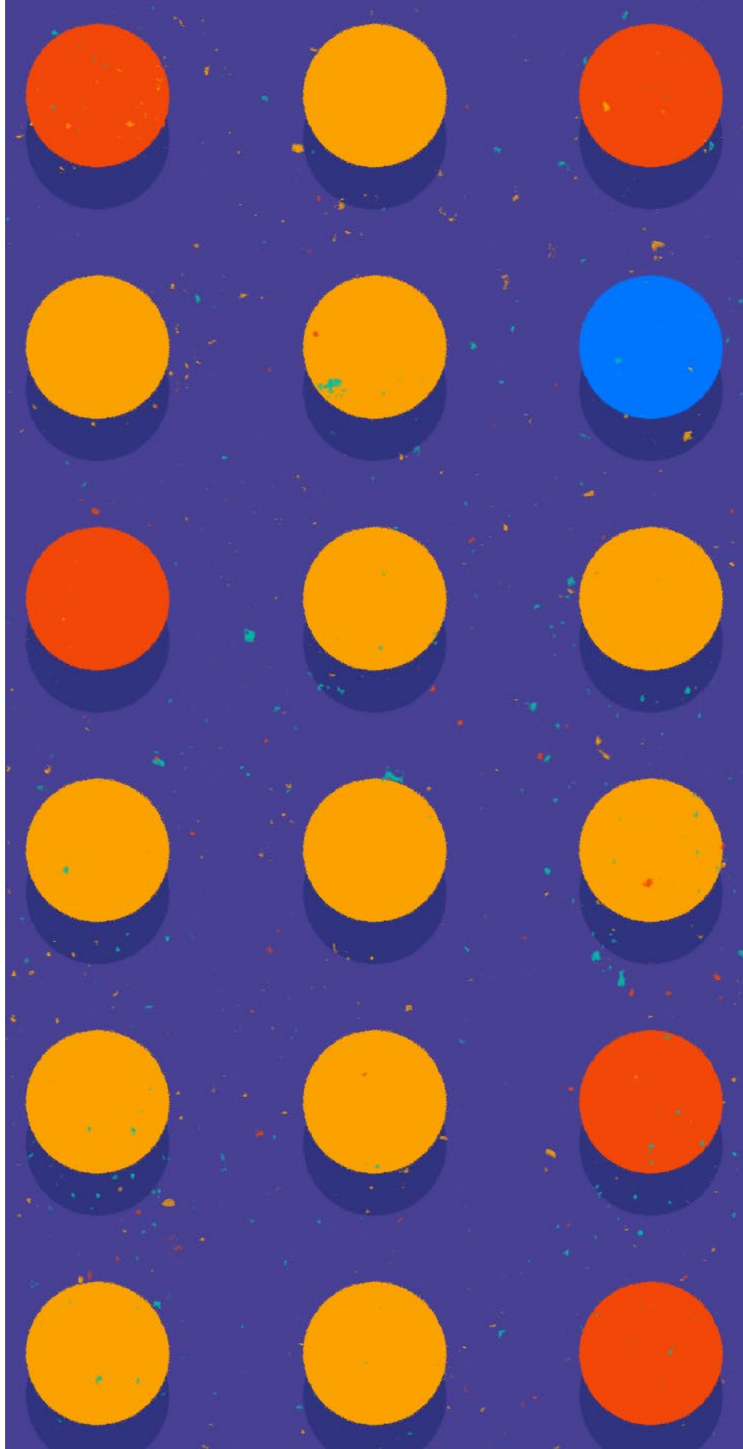
What Does OpenTelemetry mean for Observability?

Software observability can be improved with traditional telemetry—the process of gathering the performance data of any product and communicating it to a remote location for monitoring and analysis. This technique is commonly used to keep track of the performance of various products with ease. By capturing data from components, and using additional software to measure that data, a company or individual can receive valuable information and insight into the quality and abilities of the system they are using.

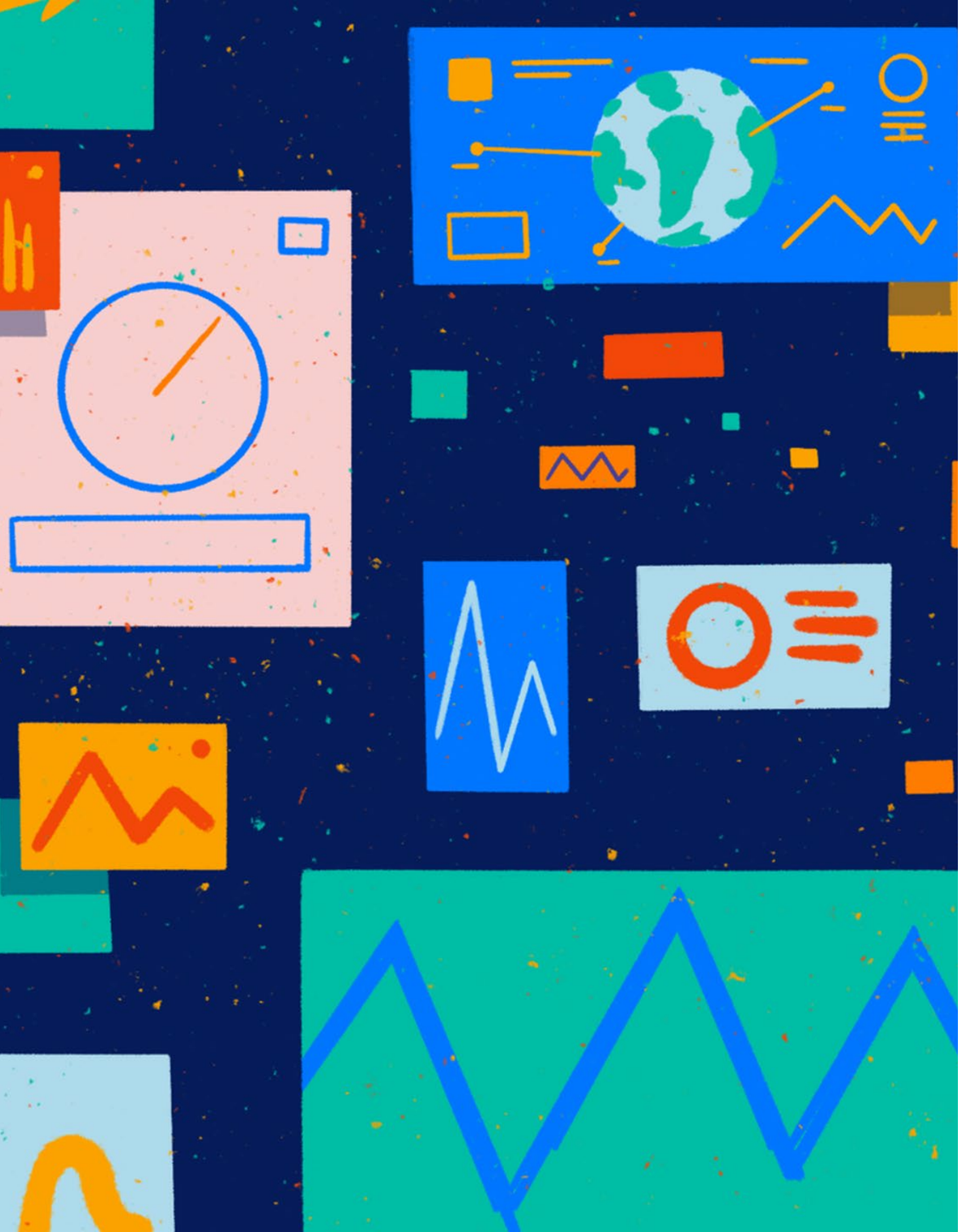
The biggest downside to traditional approaches to systems is that it requires the creation of additional code to observe and analyze that system. But with OpenTelemetry, the code is already provided and there is no need for a company to build something new that will give them additional insight. That can be extremely valuable for that company and can be used by other companies across various industries as well.

When companies attempt to write their own codes for observability, they can inadvertently introduce faults and other problems into the mix. That is an unfortunate aspect of working to create something that will analyze the system and provide insight into its internal workings. OpenTelemetry helps companies avoid all that, and addresses the issue of observability in ways that won't conflict with the system's operation.

It is easy to see that the value of OpenTelemetry is in the benefits it provides to both companies and their customers and how these benefits can be expanded upon for the future. Because OpenTelemetry is vendor-agnostic, and because it is uniform for everyone who uses it, companies and individuals alike can experience the quality it will add to their businesses and Lives.



©LogicMonitor 2021



Chapter 5: Real-time topology mapping for observability

Earlier, we talked about the three types of telemetry that are essential pillars of observability: metrics, traces, and logs. Together, these sets of data help organizations determine what's going on in a system and whether a problem exists; where the problem resides; and its root cause.

Metrics, traces, and logs provide the fundamentals of observability, but it's important not to forget that on their own, they are simply telemetry. To extract insights from these isolated data sets, we need to add context. In this chapter, we'll talk about how topology mapping provides the richer context required for observability, by capturing the relationships and dependencies for all system components across the full stack. We will also discuss how acquiring this level of context provides a foundation for AI to make a substantial impact on outcomes.

Why topology mapping matters

Metrics, traces, and logs can tell us a great deal about the health and performance of a system, and offer a record of events within it. However, an isolated observability pillar actually provides relatively little value in terms of visibility and maintenance of a system. Full observability can only be achieved by combining and correlating a tapestry of data to create a complete perspective. The relationships and dependencies between these pillars can be discovered using topology mapping.

Topology mapping is the visual representation of relationships among elements in a communications network. These maps can represent the physical location of network components, generally referred to as layer 1 mapping; the logical relationships among elements at layer 2; and additional layers, extending all the way out to layer 7.

For example, at layer 2, an intelligent full-stack observability platform utilizes the Link Layer Discovery Protocol (LLDP) as well as Cisco's proprietary version of the protocol known as Cisco Discovery Protocol (CDP) to dynamically generate network topology maps that show how data flows among the different resources like switches, hosts, firewalls, routers, and other network components in an environment.

With an additive approach to generating topology maps, only the relationships that are most relevant for the current task will be shown, but it's possible to expand the view outward. Application mapping may vary significantly from the physical or even logical network map. From an application perspective, it's important to understand the difference between users who start on a mobile app API's, compared to those on browser, even though they may be doing the same transactions.

Context may be found in seeing different cloud availability zones, or even via the tagging of the application components themselves. For example, if all slow transactions are running on a new build that has been deployed to a small number of customers, that's important context yielded from the telemetry as well. Perhaps the new build has a code problem or uses a backend database in a different way than previous builds.

Ultimately, topology helps us not only to visually map things but more importantly to figure out which things are related to one another. This allows for telemetry data to be pulled from the giant lake of data for use in examining a specific issue. The problematic application build discussed above could be associated with issues related to the database, the network, an unrelated load balancer change, new code deployed, or myriad other causes. An environment with enhanced observability could employ the guidance of an ML-driven system that suggests ITOps should look at the network logs associated with the database, but not the ones on the load balancer because they're not interesting in the context of this issue.

Shining a light on context and dependencies

Infrastructure monitoring has long been valued for its ability to enable ITOps teams to quickly determine the root cause of an incident that is impacting dependent resources. It can also provide the option to suppress alert notification routing for alerts that are determined to be dependent on the originating alert. Observability enables ITOps teams to broaden their insights, to solve issues that are not confined only to the network domain.

For example, users in an organization might be experiencing poor performance or other issues, but the cause is difficult to determine. Gaining observability could help the organization determine the difference in transactions by combining downstream alert suppression with an analysis of what types of users are experiencing slowness, in situations where one simple answer isn't apparent.

During an alert storm, ITOps teams may encounter multiple alerts relating to the same originating incident. Each time a metric threshold is exceeded, a notification is sent, and a trickle of notifications can quickly become a wave. Before long, they are up against a flood of notifications for every resource that's affected by the incident— without having a clear idea of which resources are its true cause.

Here's how gaining observability to support root cause analysis addresses this issue:

1. Identify unreachable alerts for resources in a dependency chain. Root cause analysis is based on topology relationships. If a resource that is part of an identified dependency chain goes down or becomes unreachable, its alerts are flagged for root cause analysis.

2. Delay routing of alert notifications.
When a resource in the dependency chain goes down or becomes unreachable, this first "reachability" alert triggers all resources in the chain to enter a delayed notification state. When enabled, this state stops alert notifications from being immediately routed, which buys time for the incident to fully manifest. It also frees up time to enable the root cause analysis algorithm to determine the originating causes and the dependent causes.

3. Add dependency role metadata to alerts.
In this step, any resource in the dependency chain with a reachability alert is identified as a parent node or suppressing node to its dependent child or suppressed nodes. This process adds metadata to the alert, and specifies whether its role is originating or dependent. This role provides the data that an AIOPs platform could utilize to suppress dependent alert notifications. Root cause analysis can also put the right log event in the context of an alert, automatically, to further enrich responses.

4. Suppress routing of alert notifications. For this optional function, alerts identified as dependent are not routed. This enables RCA to reduce alert noise reporting only those alerts that identify root cause.

5. Clear alerts across dependency chain.
When the originating reachability alerts begin to clear, all resources in the dependency chain are once again placed into a delayed notification state. This allows time for the entire incident to clear. After a few minutes, any remaining alerts will then be routed for notification. If some resources are still unreachable, RCA initiates a new root cause analysis incident, and the process repeats.

6. Analysis of high-cardinality problems.
Working from a streamlined set of alerts, ITOps can examine individual transactions related to the issue. Perhaps slow systems are affecting only certain users because only some nodes are resource -starved due to Kubernetes configuration issues. Observability provides the broader context needed to perform long-tail analysis to solve more complex issues.

©LogicMonitor 2021

Benefits for IT teams

For IT teams, automated root cause analysis based on AI and observability is an essential capability for saving time and separating signal from noise in alert notifications. Every day, analysts, system engineers, and architects are grappling with complex on-prem, private, and public cloud environments. They're doing all they can just to keep up, and any relief they can have from putting out fires means more time they can spend focusing on their customers or working with IT leaders and business stakeholders on more strategic projects. Root cause analysis can help organizations accelerate mean time to resolution (MTTR), and minimize alert fatigue.

Speeding mean time to resolution

IT teams know that understanding the context of an IT incident can provide the insight needed to greatly reduce the MTTR—and enhance the ability to determine the root cause. In some situations, this context might be the downstream dependencies after a high availability pair of firewalls goes offline.

In other scenarios, the context might be the datastore in contention from multiple VMs. No matter what the issue, context offers critical information necessary to understand the relevant scope when an incident occurs.

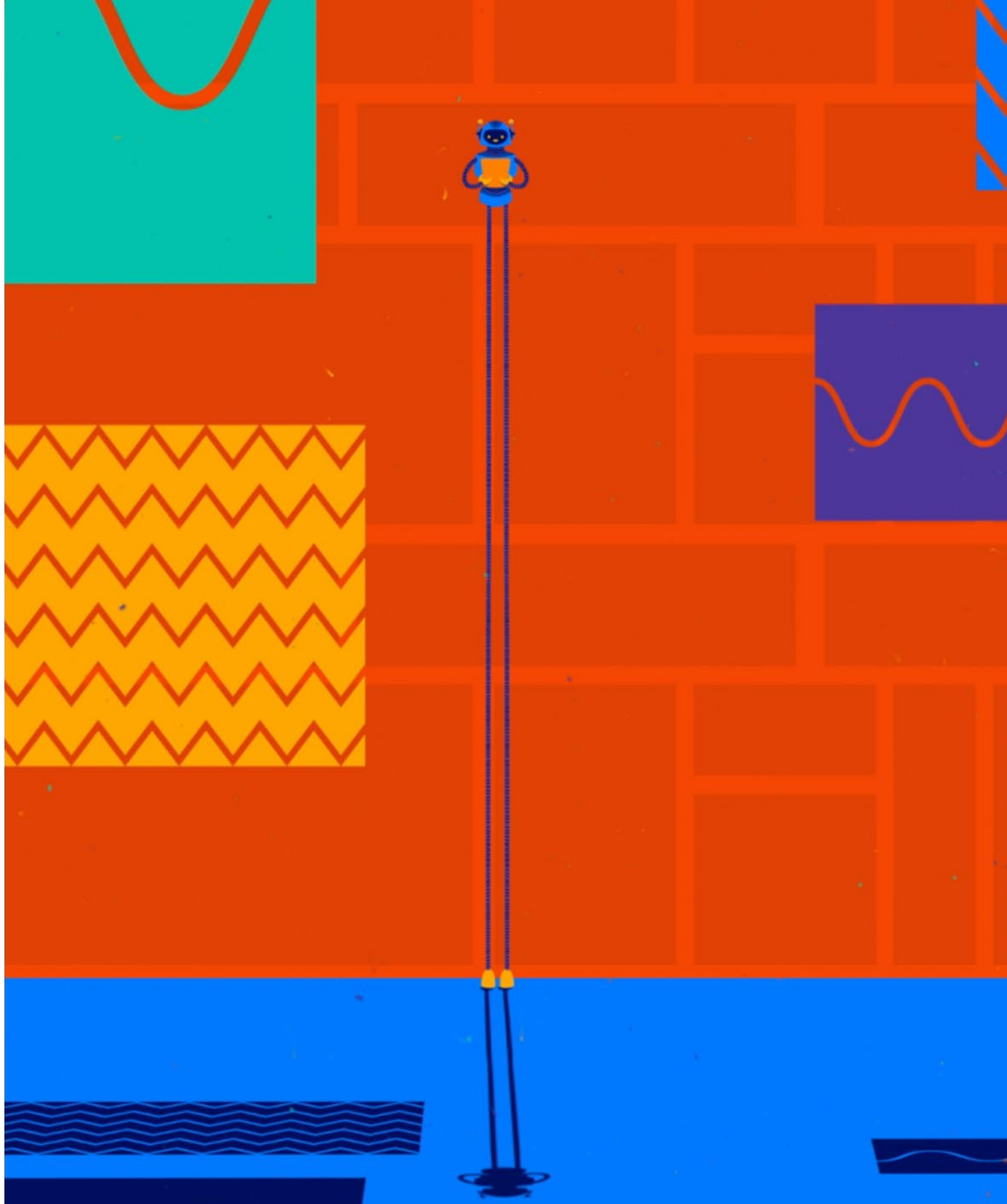
The topology mapping functionality that's part of root cause analysis gives IT teams the full context needed to troubleshoot events or incidents that may have occurred, and do it more quickly. Log analysis can also provide valuable insights into how issues are occurring, to speed issue resolution. That means more uptime for critical business systems and processes, and more time for IT to focus on more valuable tasks.

Minimizing alert fatigue

If an IT team is spending most of its time chasing down irrelevant alerts, employee satisfaction drops, frustration sets in, and communication in the team can falter, making it less effective. Alert fatigue can also result in a tremendous opportunity cost. When IT is exhausted by the alerts process, they have that much less time to come up with new ideas, offer a better customer experience, or take proactive steps to prevent future issues.

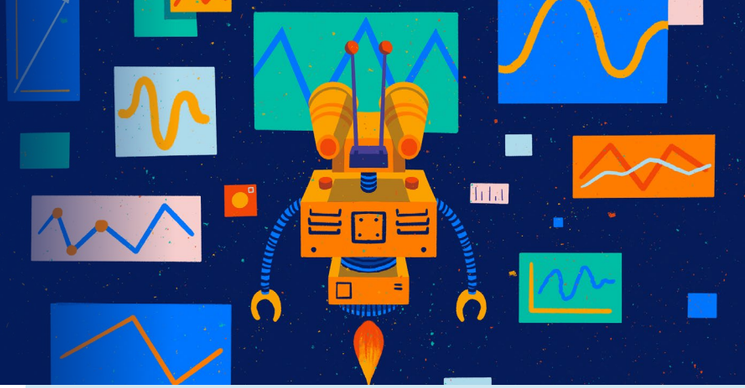
An observable environment enables AI-driven root cause analysis to surface actionable alerts within an early warning system. By reporting only on the primary cause of issues, and screening out irrelevant data according to the selected criteria chosen, it rapidly surfaces any anomalies and frees staff from time-consuming, manual work of sorting out what's important and what is not.

©LogicMonitor 2021



Chapter 6: Using AI & ML to achieve observability

As we discussed in our root cause analysis use case, AI and ML can play an important role in providing context that's key to realizing a more observable environment.



AIOps builds on observability and automation to enable organizations to support increasingly complex environments. It combines massive amounts of data with context, together with automation and playbooks that drive action. It also helps organizations ensure that the actions are netting the benefits sought, and incorporates lessons learned into the next round, using closed-loop feedback that connotes learning from previous behavior and decisions to make future decisions better.

ML/AI algorithms help organizations get out in front of issues, by automatically detecting anomalies, such as change or capacity issues in infrastructure, application, or service before they become problems. Automation plays a key role in enabling AIOps to find anomalies, build insights, and respond to them. An early warning system is fundamental to driving AIOps, by showing what is happening, where it is happening, and why it is happening.

AIOps is not only about issue mitigation, but also about continuous optimization. Its capabilities can provide support for a failure prevention system that continually learns about the environment, to optimize the technology stack so that it drives business outcomes. With an AIOps approach, IT professionals can free themselves from focusing on baseline "keeping the lights on" responsibilities and assume a more strategic role in driving innovation and digital transformation. It's important to point out that AI and automation are not about replacing individuals, but about moving people up the value stack so they can do more rewarding, engaging work. It enables organizations to focus on value creation and optimization, rather than workforce reduction.

AIOps was created specifically to address the challenges of today's modern hybrid infrastructures, and enable businesses to see what's coming, spend less time troubleshooting, and more time innovating. It lets organizations break the endless cycle of reactive monitoring to embrace observability and get out in front of infrastructure issues, moving from a proactive to a predictive approach before they can impact business operations and the customer experience.

Its key capabilities include:

- Ingesting data from multiple sources, agnostic to source or vendor, to provide a comprehensive picture of IT environments, to minimize vendor lock-in
- Enabling creation of future-proof instrumentation models that make it more modular and less risky to insert a new type of analysis later, without needing to start the whole stack from scratch
- Performing real-time analysis at the point of ingestion to open up faster insights and enable more proactive responses to issues
- Performing historical analysis of stored data to better inform insights with additional context
- Leveraging machine learning to surface the most relevant alerts, and suppress redundant data that can distract and slow IT responsiveness
- Employing automation to initiate an action or next step based on insights and analytics, to enable rapid responses to issues before they impact the business

"By embracing AI augmented automation, IT teams can better learn the skills of AI and position themselves to have more effective partnerships with peripheral business units. In fact, by 2023, 40% of infrastructure and operations (I&O) teams will use AI-augmented automation in large enterprises, resulting in higher IT productivity with greater agility and scalability."[1]

[1] Smarter With Gartner, Gartner Predicts the Future of AI Technologies, February 2020, (https://www.gartner.com/smarterwithgartner/gartner-predicts-the-future-of-ai-technologies)

©LogicMonitor 2021

Why AIOps and why now?

Why are AI, automation, and ML now making these types of benefits possible? Until recently, organizations lacked the technical plumbing to allow the type of scaled parallel processing needed to analyze immense volumes of observability data. Analysis of data at this scale simply wasn't realistic, and efforts to achieve it led to unacceptable down-sampling and fidelity loss. At the same time, there was so much proprietary telemetry data in play that translation wasn't feasible.

As we discussed earlier, today's organizations are embracing open standards, and the power of the cloud allows the promises of AIOps to be much more realistic. When combined with the richness of observability-based telemetry data, organizations finally have all the ingredients needed to truly disrupt the market—even though the problems aren't new.

AIOps helps IT deal with increasing technical complexity, as well as support business outcomes efficiently and effectively through technology. It helps ITOps teams handle their primary responsibilities more effectively while enabling them to better align their activities with key business stakeholders.

Full stack observability contributes to an early warning system

The foundation of AIOps for monitoring is an early warning system. Observability contributes to an early warning system by providing access to more data, with the ability to analyze it more thoroughly. This lets organizations see signals that weren't previously visible and thus can identify trends before they become user-impacting issues. These early warnings keep IT better informed, by employing automatic anomaly detection via ML or powerful algorithms that detect issues and their root cause, before any widespread business impact.

At the heart of the early warning system are AI and Machine Learning (ML) algorithms that support anomaly detection in any data set, such as IT infrastructure metric and log data; dynamic thresholds; root cause analysis; and automatic correlation. Working together, these features sift through massive amounts of monitored data, such as metric and log events, and surface the most important information, then make it more actionable by adding context.

These algorithms can also support advanced log analysis, where contextual and actionable log data is correlated with metrics and alerts to provide deep insight that goes beyond basic notifications, to provide insight into why issues are occurring. An AIOps early warning system helps organizations prevent major business-impacting incidents, by processing the signal through a rule-based action engine, tied into a robust automation framework. The result is a more informed, proactive IT operations team, and shorter MTTR.

Minimizing down time

One of the most compelling benefits an early warning system brings to observability is the ability to minimize downtime. When IT teams are spending most of their time tracking down root causes and putting out fires, they do not have the time or resources to switch to a proactive model where the focus is on avoiding outages instead of minimizing their impact.

Resilient, ephemeral architectures like Kubernetes give organizations the flexibility to shift gears in ways that simply weren't doable before, but with so many moving pieces, it's critical to have a strong observability practice in place to keep track of all the moving parts—and their relationships and implications for one another.

An early warning system that supports AIOps and full-stack observability will provide IT operations with the information they need to make the switch to proactively preventing problems, instead of reacting to them. It also solves the problem of traditional monitoring approaches that focus on static alerting and analysis configurations such as static thresholds. As environments have become dynamic and distributed, these types of traditional approaches weren't practical, because they required human intelligence and significant time and effort—something that today's digital businesses can't afford.

Keeping alert fatigue in check

With its ability to surface the most relevant alerts from a broad array of data, an early warning system classifies alerts and workloads more effectively. Using dynamic thresholds, it detects the normal performance range for technical and business metrics and generates alerts based on anomalies. It then alerts IT teams based on historical performance and advanced algorithms. An early warning system helps businesses avoid alert fatigue, save time, and surface anomalies sooner.

As job responsibilities evolve to match an observability-centric world, SREs and other professionals will be increasingly focused on not letting the same problem happen twice. Their role will be to ensure the right telemetry, analysis, and automation help foster a one-time-only culture around issues.

Driving a proactive approach, instead of reactive respose

An AIOps early warning system is the game-changer that lets organizations break the cycle of constantly chasing down issues after they happen. By providing the observability needed to prevent issues from becoming widespread in their business impact, an early warning system makes the flood of data produced by infrastructure environments more manageable, actionable and frees teams to focus on the big picture.

©LogicMonitor 2021

Discovering exceptions with dynamic thresholds

An early warning system is about much more than simply monitoring complex, distributed, and ephemeral environments and alerting in context. To be really effective, organizations need to determine which data points are most important, separate them from those less relevant, and notify the IT team about them, before they impact performance or availability. That process starts with improving focus utilizing dynamic thresholds.

Dynamic thresholds are based on AI/ML-based algorithms focusing on anomaly detection based on rate of change and seasonality, along with algorithms to contextualize issues. These algorithms automatically detect the normal performance range for any metric—whether it's a technical or business metric—and accurately alert based on values outside of this range that are considered anomalies. While dynamic thresholds are in and of themselves not new at all, their application to rich telemetry data allows them to become repurposed and even more meaningful.

Because dynamic thresholds (and their resulting alerts) are automatically and algorithmically determined based on the history of a datapoint, they are well suited for datapoints where static thresholds are hard to identify, such as monitoring the number of connections, latency, and other criteria. Dynamic thresholds are also useful in situations where acceptable datapoint values aren't necessarily uniform across an environment.

As cloud deployments increase and environments become more volatile, dynamic thresholds also provide a more effective way to discover exceptions in these ephemeral environments. While static thresholds are too difficult and time-consuming for people to manage manually, dynamic thresholds provide an opportunity for AI/ML-based algorithms to demonstrate their advantages in fast-changing environments.

Identifying patterns and pattern types

When most people hear about dynamic thresholds in the context of monitoring, they think of the ability to simply trigger alerts based on a non-static threshold. Although this is a great capability, it's only half the picture.

An effective solution will do more than simply trigger alerts by identifying issues that wouldn't be caught by traditional static thresholds. It can also help IT teams focus, by eliminating excess alerts caused by thresholds that haven't been tuned well, surfacing the important issues. This lets them use dynamic thresholds as a filter to reduce alerts to only what is immediately relevant, reducing noise and enhancing observability.

Simply put, dynamic thresholds represent the bounds of an expected performance range for a particular datapoint. Even the bounds themselves are subject to change—for example, a CPU running at 85% capacity might be abnormal for certain workloads, but normal for others. For example, observability data could reveal that an unusual workload is driven by higher traffic related to a new marketing campaign launch, rather than a denial-of-service (DDoS) attack. Unlike static thresholds that are assigned manually, dynamic thresholds are calculated by anomaly detection algorithms and continuously trained by a datapoint's recent historical values.

Dynamic thresholds apply algorithms that calculate the thresholds to be set and continually adapt to environmental changes, both from a technical and business perspective. In other words, alerts are generated when anomalous performance is detected. These dynamic threshold algorithms can detect a variety of different types of data patterns, including:

Anomaly detection — To discover anomalies, historical performance is used to generate an expected range for resources. This range is used as the basis to highlight and alert on activity that breaches this range and is considered anomalous. With anomaly detection, activity that may have previously triggered an alert is now silenced based on historical performance.

Rate of change — Anomaly detection is not informed by metric value alone, but also enables dynamic thresholds to alert on anomalies in metric value rate of change. For example, monitoring could produce a significant increase in a metric that has not yet crossed a static threshold, such as a disk that is rapidly filling up. Providing insight into a metric's rate of change enables IT teams to catch issues sooner before they negatively impact the business.

Seasonality — Dynamic thresholds can map seasonality to business processes. Algorithms ensure that an AIOps solution identifies patterns in resource performance, and alerts on anomalies in these patterns. For example, an organization might have a VM in place that normally backs up on a daily basis but has failed to back up on a particular day. An early warning system could alert them to the issue right away, well before it becomes a more serious issue.

A robust AIOps solution can also present an auto-generated range that is widely applicable with sensible defaults while supporting customization with advanced configuration options.

Dynamic thresholds enable teams to understand the correct level of expected performance—and immediately spot where it deviates from normal and requires attention. At the same time, they help ensure that teams aren't receiving irrelevant alerts for optimized machines that are regularly highly utilized.

In situations where it is important to determine if a returned metric is anomalous, dynamic thresholds have tremendous value. Not only will they trigger more accurate alerts, but in many cases, issues are caught sooner, before widespread business impact. In addition, administrative effort is reduced considerably because dynamic thresholds require neither manual upfront configuration nor ongoing tuning.

©LogicMonitor 2021

A more successful, collaborative IT team

Empowering ITOps with observability and AIOps enables them to do their jobs more proactively while bringing them closer to business imperatives. By providing the relevant data that ops teams need to make better decisions, faster, they can unleash service improvements that drive compelling business outcomes—which are made possible only through a foundation of observability.

Maximizing service quality and availability

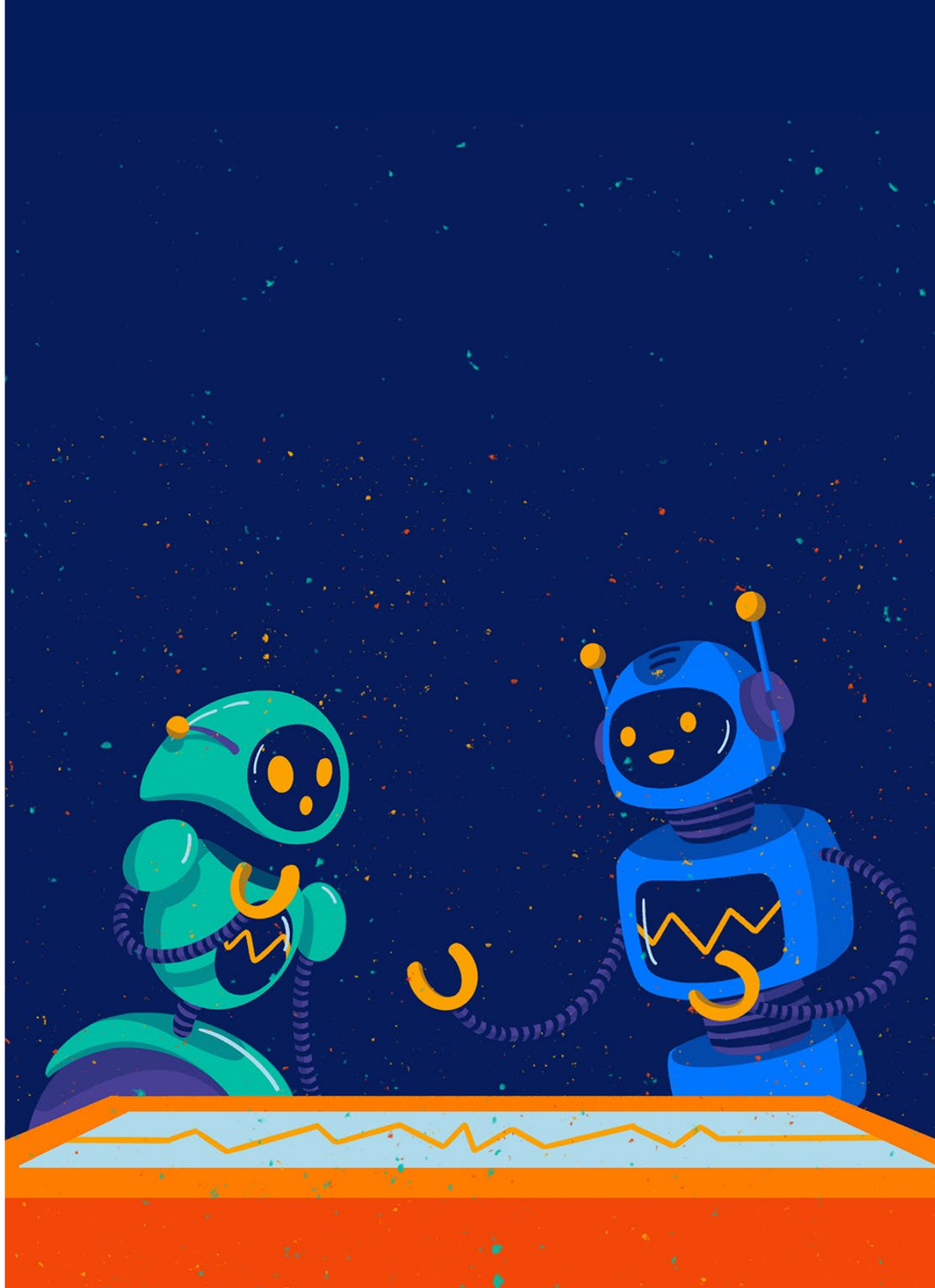
AIOps gives IT teams not only the visibility and insight they require to spot potential issues, but the context and automation they need to prevent incidents. For example, instead of waiting for an outage to occur, AIOps can identify troublesome resource usage patterns that could cause an outage if they continue. After discovering the potential problem, AIOps could deliver actionable recommendations to avoid the outage. When integrated with an automation platform, AIOps lets organizations address the issue even more quickly, executing recommendations and fixing resource usage well before a problem can escalate and impact users.

AIOps can also deliver valuable insights needed to support more stable, highly available customer-facing services. For example, observability could help an MSP service delivery manager understand which applications and infrastructure issues are most likely to impact the stability of their customer's environment—and the user experience—and take steps to prioritize and eliminate them. The result is a superior experience, improved user satisfaction, better long-term customer retention, and enhanced revenues.

A team mindset for problem-solving

As data becomes more essential to every organization, business and technical teams are becoming more collaborative. Technical teams are looking to align better with business imperatives, while business stakeholders seek metrics to understand how technical issues affect their priorities.

With full observability and AIOps, organizations can apply contextualized data to create more precise, tangible insight into how an incident will impact operations, and end customers. Instead of collecting and analyzing metric, log, and application data from several systems, infrastructure, operations, DevOps teams, and other business stakeholders can easily access and visualize insights and get answers in the context of their role.



©LogicMonitor 2021

Chapter 7: Scaling your IT to achieve observability

The advantages of improved observability are clear, but today's highly complex architectures and services introduce some distinct challenges to achieving it. First of all, observability isn't prioritized when monitoring tools can't scale with their containers. That means being able to scale literally everything is crucial to implementing observability.

In this chapter, we'll take a closer look at the role of scalability in observability, and the importance of scaling IT to ensure it. We will present some approaches to address business and organizational challenges, and offer recommendations on scaling time, cost, and data collection and analysis.

Complexity of modern environments is soaring

Competition is growing more intense in every industry, and organizations are doing all they can to keep pace with the accelerating pace of business. Organizations are aggressively embracing microservices architectures, which provide valuable advantages in agility, modularity, and flexibility they need. However, these modern architectural approaches also introduce new complexities that make observability challenging.

Cloud infrastructures and application services are composed of dynamic, short-lived hosts. The data they generate is ephemeral and often voluminous, and the pace of development and IT is accelerating. If data is not collected in real time, it's difficult to discover and identify an issue, troubleshoot and understand it, and take steps to mitigate its impact.

"Organizations are rapidly adopting multicloud, containers and microservices. Yet, they struggle to migrate their existing monitoring infrastructure to ephemeral architectures, leaving frustrated end users and large gaps in visibility."[2]

The sheer volume of data is also challenging. Containers, cloud platforms, and microservices constantly create huge volumes of telemetry and data. This hampers the ability to digest a huge amount of data and bring it all together in a way that's really useful to help teams reach the conclusions they need to proactively address issues and speed mean time to repair. The continuous delivery and integration that these environments require also makes monitoring of this information even more challenging.

Scaling for efficiency and speed is key

Given the tremendous volume and complexity of modern services and architectures, it's clear that organizations need the ability to fully scale their analysis processes and tools in a way that will not overwhelm their teams. The process starts with a strategy to optimize for time and efficiency. Organizations should take steps to ensure that they can maximize their investments in observability and monitoring solutions, as well as the skills and talent they have on their teams.

A strong initial step is to explore ways to minimize silos of data that could hamper collaboration. Organizations should identify data that could make it difficult for DevOps and other teams to communicate within and between their organizations and share information. As they grow and their needs change, they will need monitoring and observability tools that can scale with them and keep pace with change.

Incorporating a role-based approach to collaboration and information sharing can help organizations maintain control over data, even as they make it more available. Applying role-based access control can help groups ensure that people will have access to the data they need to collaborate smoothly across silos, without compromising compliance needs.

Automation can play an important role as well in making analytics processes more efficient. For example, streamlining steps like querying can help reduce time-consuming steps related to manual analysis.

[2] Padraig Byrne, Sr. Director Analyst, Gartner, "Monitoring and Observability in a Hybrid Cloud world"

©LogicMonitor 2021

Aligning observability to business imperatives

Observability is fundamental to driving better outcomes, so it only makes sense that an observability strategy can scale in a way that aligns with an organization's business strategy. Technical capabilities and processes for observability work best when they are future-proofed. They should be designed to remain effective against potential changes in business or technical strategy that could impact them. For example, OTel is commoditizing the data-gathering layer to future proof telemetry and help organizations avoid rip-and-replace decisions that used to cost them millions.

An observability deployment should be built to consider any applications performance issues that will clearly impact user experience, as well as the team's budget. It should also be architected to prevent the organization from having its practices be overly dependent on platform licensing models or other criteria that may limit flexibility.

Observability should be integrated as part of an organization's process and culture as well, through applications and services that have telemetry architected from the beginning. Development teams should be allowed time to ensure they are instrumenting their deliverables with observability in mind. Organizational structures should reflect this focus as well, with an aim to enhance ownership rather than support "throw it over the fence" behaviors. Organizations that can instrument applications intentionally from the beginning will get much better outcomes than those that need to circle back and modify them later.

Scaling data collection and correlation effectively

To build and sustain observability in a way that successfully meets data handling challenges, organizations need processes that are designed to accommodate a significant variation or spikes in the volume of data that needs to be ingested and analyzed. For example, some organization needs may change depending on seasonality and other variables.

A strong observability solution ought to allow an organization to turn the dial of collection up and down dynamically in a way that allows collection to be always on—but not cost-prohibitive. For example, an organization may gather a baseline level of data all the time, but perhaps discard the detailed logs after extracting normalized metrics from those logs. However, when a problem is identified, the organization could retain all its logs and data, to potentially analyze, transform, and make decisions about data before it's ever written to disk. A fully mature observability implementation utilizing dynamic data collection could support stream analytics and other capabilities that filter data and augment engineering capabilities.

Observability should also be implemented to account for the requirement to handle data effectively across all of its key core functions, including ingesting, correlating, analysis, and other parts of the process. The implementation should be designed to consolidate metrics, traces, and logs to reduce the overhead of storing and monitoring data—and to minimize data silos. It should also utilize a monitoring approach that is massively scalable and flexible to support variable and demanding requirements.

The advantages of an as-a-service approach

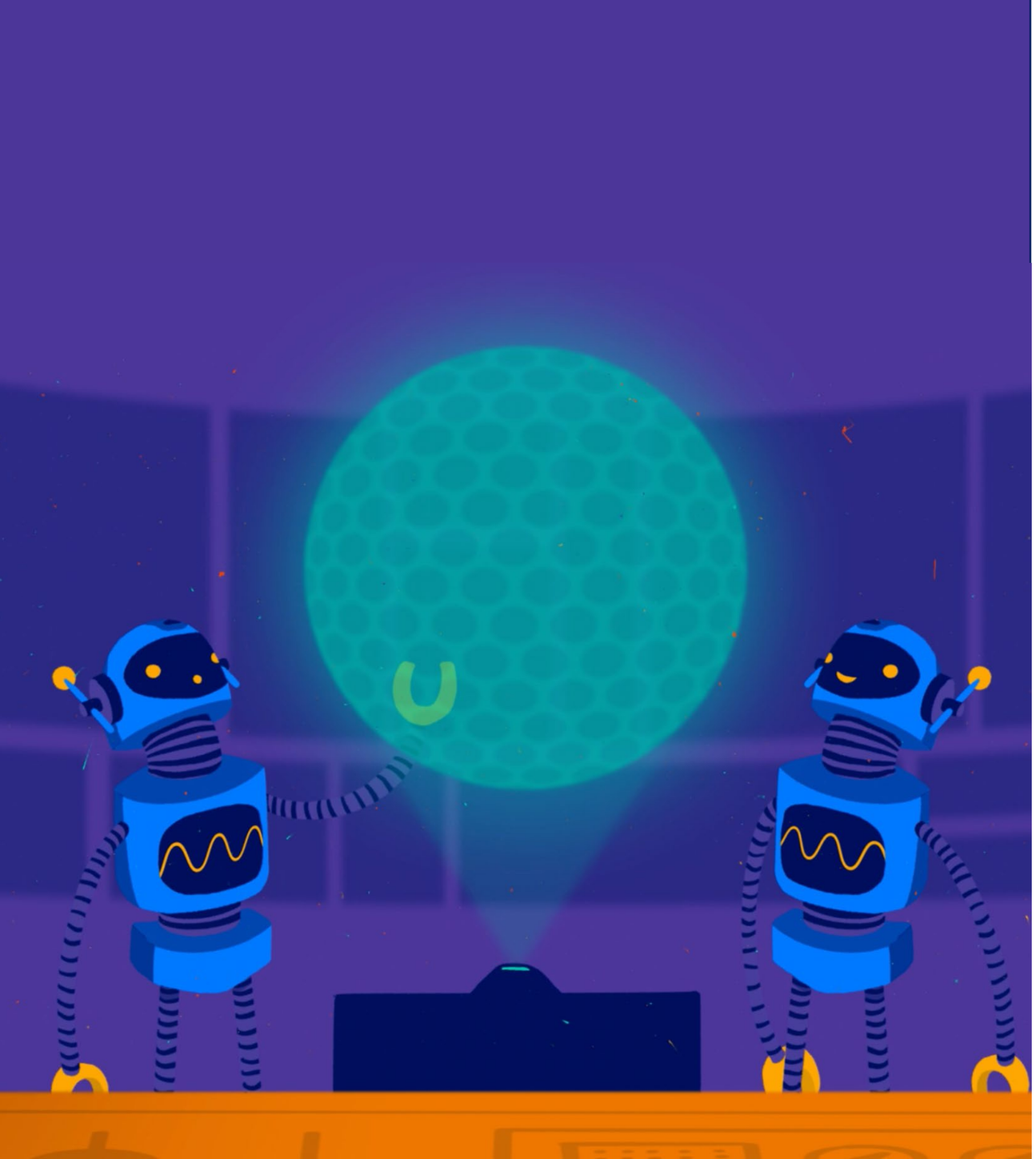
The monitoring and AIOps solutions required to support observability are evolving all the time, and new technologies and better algorithms are constantly appearing. To keep pace with constant enhancements, a software-as-a-service (SaaS) delivery model can support the scalability and flexibility that organizations require to keep pace with growth and changing needs.

As your business needs continue to change and the infrastructure evolves, an organization's monitoring and AIOps platform should also be able to scale with it. An as-a-service approach can help organizations save time and realize additional business outcomes by enabling them to extend their solution across more data types and analyzing more data. An as-a-service model also lets organizations get out of the business of managing and maintaining an observability platform. It frees their valuable professionals to spend time working on higher value-add activities and leave the platform management to the experts who can do so better and more efficiently than they could on their own.

Compared to an on-premises solution, a SaaS monitoring and AIOps platform is easy to deploy, and is regularly enhanced to take advantage of the very latest technologies. If a new algorithm emerges that can accelerate troubleshooting, strengthen efficiencies, or enhance process signaling, organizations can take advantage of it right away, instead of waiting for manual upgrades that can take time and resources to apply to an on-premises product.

In our next chapter, we will explore some of the challenges and approaches to implementing observability in a variety of use cases, including cloud scaling for GCP, Azure, and AWS, as well as Kubernetes and general containers and microservices.

©LogicMonitor 2021



Chapter 8:
Implementing observability

We've discussed the importance of scaling IT to successfully enable observability, regardless of how and when applications, software, and systems expand or shrink. However, assuring scalability is just one of many challenges to implementing an observability initiative.

Infrastructure, operations, and DevOps teams also face challenges that are specific to their use cases and workflows. They are working with increasing volumes and types of data and alerts from a variety of sources. Complexity is increasing, as organizations deploy new cloud environments, and implement dynamic containers and microservices. At the same time, DevOps and other teams often use multiple monitoring or analytics tools that don't necessarily sync with each other.

In this chapter, we will talk about how to overcome challenges for implementing observability and discuss approaches to applying it to a variety of use cases. We will show how to implement observability in:

- Application performance monitoring (APM)
- Log systems and processes
- Kubernetes and containers
- Cloud services

Although it may seem daunting for some companies to overcome these obstacles, it is possible to achieve good observability by considering efficient ways to deal with these challenges head-on.

The common challenges in achieving observability

As organizations consider how to enhance agility, they may find themselves limiting their focus to having the best metrics data, or the most complete traces, and pay individual vendors to provide them. While this can be an effective solution for organizations that have the resources and ability to collate and combine this telemetry, a better path to observability is to have all this information in one place—and a new way of thinking about how to overcome issues related to siloed data.

Ultimately, the focus on observability should be a democratized approach, based on empowering individuals with enriched data to drive better outcomes. In many cases, without appropriate context and correlation, providing only a deep dive into data can provide diminishing returns.

One way to optimize observability solutions is to ensure the right type of datastore is used. Datastores and data warehouses need to be able to expand and diversify exponentially to deal with the increasing volume of data and the various types of data streaming from a variety of sources. Extract, transform, and load (ETL) can help bring data together in a single, usable format and destination. However, it's even more essential to implement a system that supports a wide variety of diverse environments without enforcing rigid schemas. These systems should support extremely high cardinality because outliers in the long tail of data often hide the true root cause of problems. Traditional systems don't work well with high cardinality issues and must be rearchitected from the ground up to support such non-hierarchical solutions.

Another key challenge is enhancing observability in organizations with a range of cloud-based environments. Tools provided by public cloud providers are often limited to their own services, so it's worth considering utilizing advanced monitoring tools that support multi-cloud environments. Tools that are built to deal with diverse modern public, private, and hybrid cloud environments are more likely to adapt to changes within cloud environments, giving users stability and consistency of data analysis.

As we've discussed earlier, major vendors like AWS, GCP, and Azure all support the OTel initiative, which aims "to make high-quality telemetry a built-in feature of cloud-native software."

OTel can offer major potential benefits for DevOps teams investing in a cloud-based future for their apps and software. OTel bills itself as "an observability framework," providing tools, SDKs, and APIs purely for analysis of a system's performance and behavior. The objective is to provide some level of observability regardless of which third-party vendors businesses choose, and to provide low-code or no-code solutions for observability.

Implementing observability for application performance monitoring

As organizations have embraced DevOps, their application architectures have also evolved to support a faster, more nimble approach that better aligns business objectives and IT efforts. Architectures are no longer based on a monolithic technology stack, but utilize a layered approach that can include cloud environments and microservices. They are also increasingly moving toward more modular infrastructures. Application performance monitoring (APM) tools are designed to help organizations discover and address the root causes of issues in these complex systems.

APM can track the user experience, application latency, and identify bottlenecks within an application to resolve issues. However, as APM evolves, its role is expanding beyond simply monitoring. More than ever, APM should be considered as an important enabler of observability benefits. Organizations can perform APM without observability, but they cannot be successful in implementing observability without APM capabilities.

Unlocking observability benefits with APM infrastructure

Past approaches to APM used a proprietary, agent-based approach to data collection. Bytecode instrumentation resided and ran within each application, providing visibility for ITOps teams. Although traditional APM tools provided visibility without requiring a developer, they also introduced additional overhead into applications, and their instrumentation was often imprecise.

As development has evolved away from monolithic applications, toward modern microservices models, it's no longer possible to implement, scale, and manage traditional agent-based approaches for APM. Given this paradigm shift, today's APM relies more upon developers to natively instrument their code with precision. Ideally, this instrumentation step should be budgeted into development time, to help ensure that observability is baked into every application that goes into production. Organizations shouldn't be looking at APM for its own sake, but as an important component of an observability problem, meaning that traces must link together to logs and metrics. APM acts as an enabler to all the observability data. A good APM trace, together with AI and ML capabilities, can help ITOps determine exactly which logs or infrastructure metrics they might also want to examine.

A vendor neutral approach

When implementing observability for APM, developers should employ a vendor-neutral approach to writing and instrumenting code, so it can be more easily consumed by APM. The battleground is shifting from who can gather more data, to who can integrate it and draw conclusions from that data in a better way.

For organizations choosing an observability platform, integration with OTel is also important. This lets organizations take advantage of an extensive collection of OTel client libraries to instrument their application, without any proprietary formatting.

Utilizing a single platform for ITOps and DevOps built on OTel and OpenMetrics can enable a vendor-neutral approach to accelerate business transformation, while eliminating dependencies on proprietary technologies. The solution should be able to push or pull any data, from any source, and deliver real-time insights into complex distributed, cloud-native, ephemeral applications.

Acquiring context and achieving correlation

In observability, context and correlation are often more important than the numbers and types of data sources. This is what APM does for observability, because although metrics, logs, and traces play important roles, the benefit of a trace is not just to show the topology in the form of a map, but to help advise ITOps on which questions to ask, and where to troubleshoot issues. To acquire APM data, an OpenTelemetry collector can be utilized to receive traces from an application. Once the collector is stood up, users can instrument their applications to automatically capture trace data and forward that data into their observability platform.

With the right solution, trace data that is collected will be automatically mapped to resources that are already monitored by the observability platform, and new resources will be created if they do not exist. This way, users will be able to correlate application and infrastructure data in the same view and create a single pane of glass for faster troubleshooting and resolution.

©LogicMonitor 2021

Implementing observability for logs

Logging provides information about events and transactions taking place in an infrastructure or other technical environments. Without logging all events, organizations will have a difficult time identifying issues or determining the root cause of a problem.

Logs are typically unstructured, monitoring enables observability by collecting and centralizing logs and event information from a variety of technologies. The log data is aggregated over time and retained, and accessible for a defined period of time.

Among the more common log types to monitor are event logs, access and audit logs, and transactional logs:

- Event logs contain important information about what's happening in a system and help determine an underlying issue.
- Access and audit logs keep track of who's accessing a system and keeping track of their activity.
- Transactional logs show how systems are interacting with each other, like a web server connecting to a database server.

Unlike APM, log analysis focuses on collecting and aggregating logs from a variety of sources, making it easier to search across a broader variety of systems. APM focuses on application performance and is used to identify user experience and ways to optimize the application.

Limitations of logs alone

Logs are typically unstructured, in that organizations don't know what's in them, and traditional log tools were capable of asking any question at any time of any data. However, this capability is not entirely practical for observability challenges, where organizations are seeking to bring in all the metrics, traces, and logs from all their systems—on-prem, in the cloud layer, at the app layer, and the infrastructure. In these cases, they may not know which questions they want to ask, or where they want to ask it. The log problem is present and relevant at all layers of the equation. It is most obviously associated with infrastructure, but just as prevalent for applications, as well as business outcomes themselves. Logs remain valuable, but they are often a needle-in-a-haystack proposition that makes it difficult for organizations to find what they are looking for.

Adding value to logs for observability

In the observability arena, logs are less about investigation and more about analytics. The latest logging tools are less about asking questions, and more about being easy to use for a DevOps end-user. These tools focus on telling them things they should know and making it easy to drag and drop and correlate the metrics and logs and traces together.

Modern log tools are increasingly focused on developing metrics to evaluate the value of logs. Instead of forcing organizations to evaluate massive amounts of data, the organizations can store normal transactions and exceptions as a metric value, rather than storing all the unstructured data. This lets organizations reduce the cost of what they are storing and the time to access it down dramatically. With stream processing, they can retain only issues and exceptions. They analyze the data before it ever gets written to disk, and make decisions related before it ever gets put to the repository for queries.

Effective logging in observability is less about ensuring that a tool can acquire every possible format, and more about ensuring that they can utilize tagging to support analytics. These analytics can inform organizations which events are happening the most frequently or the least frequently—or have never been seen before. When joined with metrics and traces, they can help organizations piece together and correlate the full story and context related to an issue.

As logging tools increasingly support simpler operations, they make it easier for an average DevOps professional using them, along with tagging and ML to identify outliers. Logging in observability can help organizations better identify the "unknown unknowns" that they should be examining.

©LogicMonitor 2021

Implementing observability for Kubernetes

To keep containerized applications healthy, organizations need insight into the performance of the applications that are running in the containers, as well as the health of the underlying container resources and orchestrating components.

Containers and pods are ephemeral resources, and it's difficult to monitor a component that may appear or disappear quickly and unpredictably. As with APM, organizations should build time and budget into their development processes to practice the principles of observability. If issues arise, they will have prepared the environment so that they can identify and triage and diagnose that problem, without going into crisis mode. Implementing observability within the DevOps process can also help organizations incentivize better code quality and resilience.

One effective approach to accurately track ephemeral resources is to employ event-based discovery, where cluster resources are automatically added and removed from monitoring based on Kubernetes events or changes in microservices. A robust solution will enable organizations to dynamically group together containers and other ephemeral resources supporting a common microservice, to focus on the health and performance of the overall service, regardless of changes in underlying resources.

A single platform for rapid deployment

To gain observability into their containerized applications, organizations need the ability to place performance insights alongside their organization's existing monitored hybrid infrastructure in a platform that enables context and correlation. The solution should pull in metrics, logs, and traces from the lowest layer of the stack to the highest, on-premises and in the cloud, across all the different hops that an app goes through, and show them together in a unified view.

The solution should automatically discover containers, microservices, and underlying resources, to help save their teams valuable time, without having to worry about whether their monitoring is keeping up.

The solution should provide the ability to:

- Comprehensively monitor Kubernetes, from clusters to applications to overall health.
- Support built-in Docker integrations to make monitoring Docker containers easy.
- Provide monitoring and pre-configured alert thresholds for AWS Container Services, such as EKS and ECS.

Tracking performance and SLAs

With the right observability solution, organizations can gain deeper insights with customizable dashboards and reports that create meaningful views for your container and microservices environments. They can also share at-a-glance views of their container and microservices environment with stakeholders to keep them informed.

It can be challenging to maintain data continuity with the complexity of modern applications. A good solution will be able to utilize data across resources supporting a common application to provide long-term views into how that application performs over time. Key performance indicators are aggregated across these grouped resources to provide monitoring for the health and performance of the service as a whole.

A robust observability platform will also calculate SLA conformance with data that reflects the health of the overall service, instead of calculating based on the availability of individual underlying resources. This helps organizations be sure that they are measuring the performance criteria that matter most to their business outcomes.

Strategic planning with data retention

A strong observability solution will enable organizations to not only track, monitor, and issue alerts on containers and microservices, but also retain that data for months or years. Robust data retention capabilities will enable organizations to identify trends and forecast utilization over time.

Implementing observability for cloud monitoring

Cloud monitoring is the process of reviewing and managing the operational workflow that makes up a cloud-based IT infrastructure. Different techniques confirm the availability and performance of websites, servers, applications, and other cloud infrastructure.

A comprehensive monitoring strategy can help organizations gain the observability they require to optimize and maintain overall cloud performance, leveraging data to drive better business outcomes. It gives users real-time, data-driven insight into every potentially impactful component of their cloud deployment.

Acquiring a holistic view

A robust observability solution should be a vendor-agnostic one that can support a variety of cloud services such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP), and SaaS apps such as Microsoft Office 365, Salesforce, Zoom, or others. Monitored data for these cloud environments can be presented in auto-generated dashboards alongside any monitored data for on-premises infrastructure. The solution should enable organizations to bring their multiple cloud environments together into a holistic view.

An observability solution collects data via an application program interface (API) for individual cloud providers. These collectors are presented within the AIOps platform the same way as metrics are for all resources and device metrics.

©LogicMonitor 2021

Rapid configuration for expedited outcomes

To expedite observability, organizations should choose a solution that provides a fast setup wizard that automatically discovers, applies, and scales monitoring for their entire cloud ecosystem.

The solution should offer rapid API-based monitoring of business-critical cloud platforms and SaaS applications without deploying any agents or collectors. It should offer executive-level dashboards and deep-dive technical insights into AWS, GCP, VMware, and Microsoft Azure together with other infrastructure on one unified platform. The unified platform should also support enhanced visualization capabilities for on-prem, cloud, and microservice topology; streamlining previous workflows that may have leveraged legacy or vendor-agnostic platforms.

Reducing MTTR with intelligent alerting

Dynamic cloud environments can be hard to manage, especially when it comes to alerting. With AIOps, users can leverage dynamic thresholds and anomaly detection to intelligently detect signals from noise. Alerts are sent for abnormal behavior, helping businesses avoid alert fatigue and surface anomalies sooner. Advanced algorithms detect log events representing change and anomalies across entire environments, allowing ITOps teams to focus on the right information at the right time

Complete view of cloud provider availability

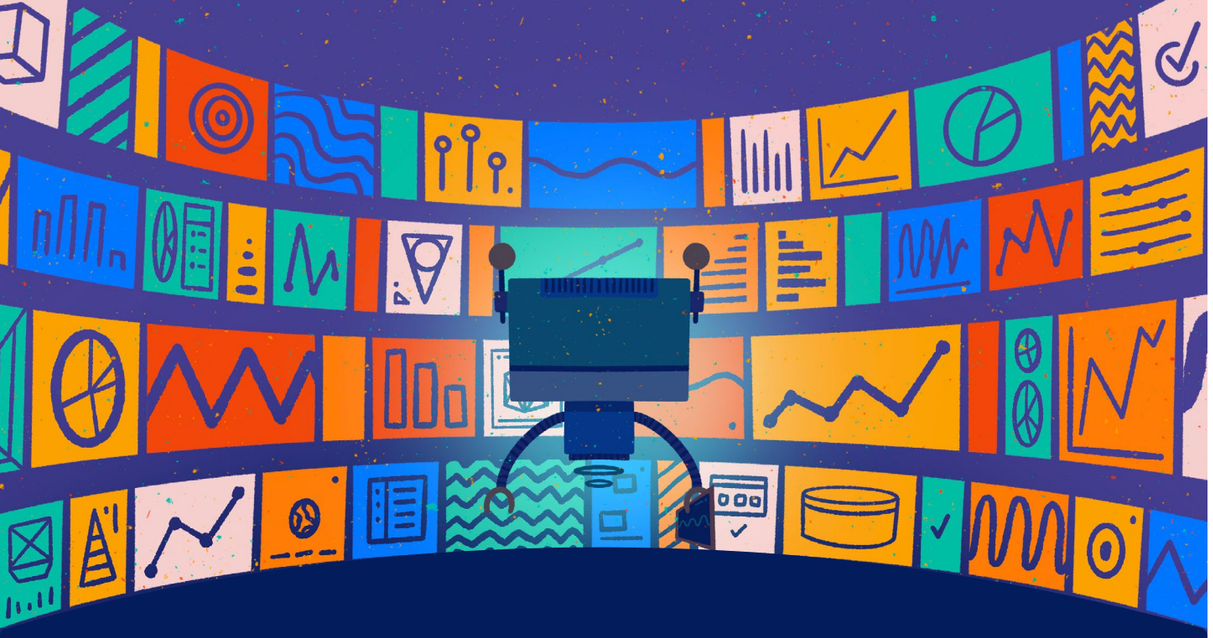
Cloud provider monitoring can provide one view into account level capacity, scheduled maintenance, and available services. This ensures teams will not run into service limits, and can forecast accordingly.

Providing full observability into public cloud platforms and SaaS applications helps organizations troubleshoot quickly and better understand application performance. It also enables organizations to correlate configuration and performance changes, using AIOps and Log Analysis to proactively identify anomalies that impact performance.

Getting the most out of cloud

Enhancing cloud observability helps organizations minimize surprises because they can quickly see at a glance where they can cut costs and optimize spend with detailed ROI analysis. Automated alerts flag spend thresholds and reserved instance expirations. They can also make operational expenses more predictable by using historical data to forecast future spend and optimize resource allocation.

©LogicMonitor 2021



Chapter 9: How close are you to full observability?

As we've considered what's required to achieve observability, and some of the basic steps involved in implementing it, one key takeaway is that that technology is only part of what's required.

There's no doubt that the three pillars of observability, metrics, traces, and logs, are important. However, they are not the end goal, but sources of telemetry for achieving observability. In fact, organizations can also utilize a variety of other data sources to help achieve observability, including indicators such as Service Level Agreements (SLAs), Service Level Objectives (SLOs), and Service Level Indicators (SLIs).

Complex systems rely on effective monitoring tools that are built with cloud-based environments in mind — but utilizing these tools does not guarantee observability, as observability is a holistic concept encompassing entire systems.

In 2020, IT leaders surveyed substantially increased their investments in data security (73%), cloud technologies and services (71%), and IT automation (68%)]. The survey also showed that planned enterprise IT investments and priorities for 2021 and 2022 reflect the new realities of today's post-COVID world, such as near-universal remote workforces and digital-first business transactions. The observability solutions that any organization invests in should be adaptable and scalable to grow with their business.

In this chapter, we'll discuss how organizations can assess their progress toward observability, and start building a strategy to accelerate their journey.

Establishing a baseline for progress

As we've outlined in our implementation discussion, observability is not necessarily a final destination, but should be considered as a culture that can be promoted within an organization on an ongoing basis. An observability mindset can start with DevOps professionals as they instrument code for APM, extending all the way out to ITOps professionals managing hybrid and multi-cloud environments.

The first element of the strategy is to determine the organization's current status. Consider how well you have defined your organization's desired outcomes, and how they are reflected in your current data architecture. Evaluate your data architecture and data workflow to determine how well they support your organization in driving toward your business goals.

This definition process will help you identify the best solutions for your specific data architecture, business needs, and company culture.

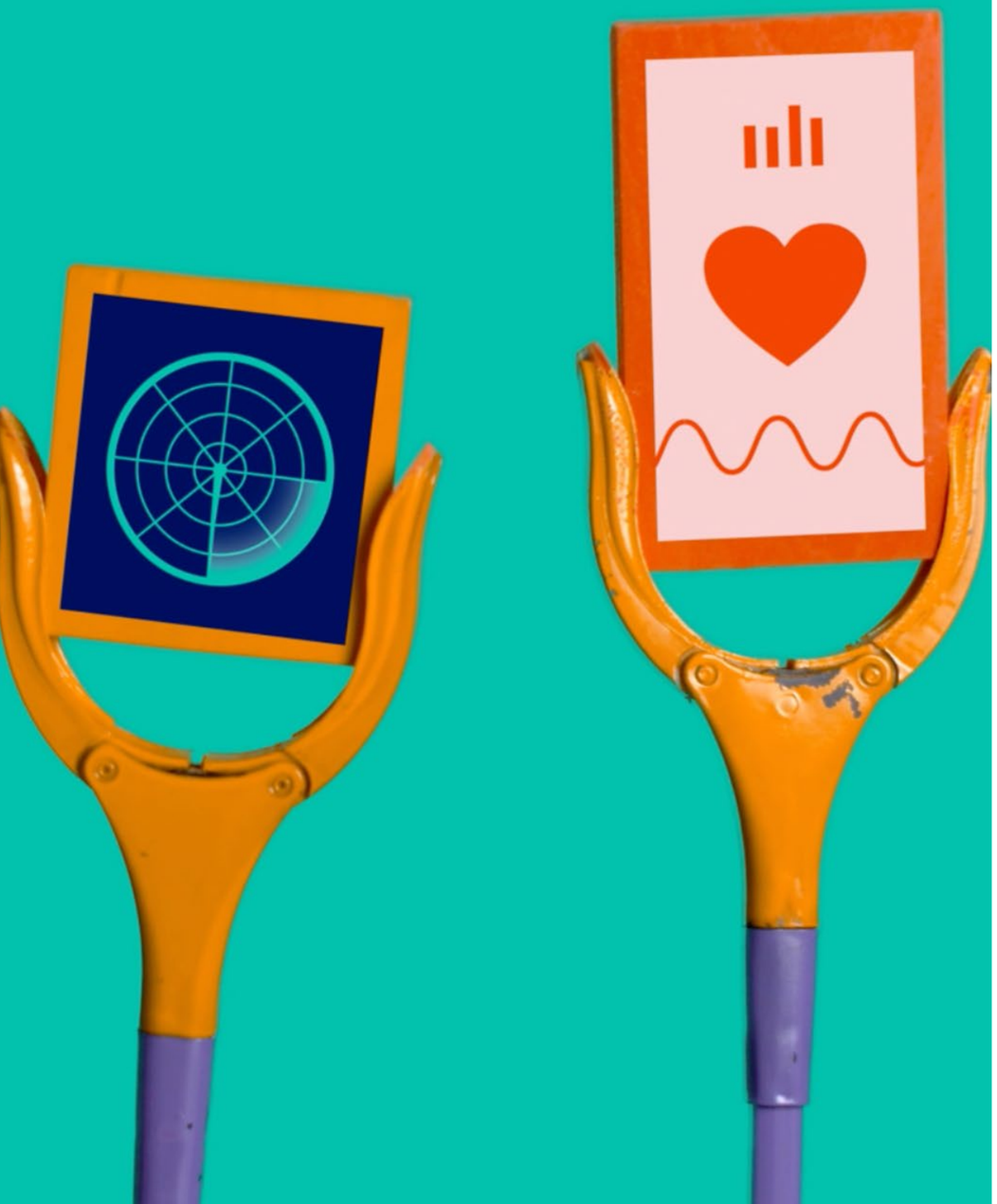
Next, you'll want to bring together all of the data that you're collecting across disparate data points. You will be gathering performance data from every environment, system, and component of your architecture. As we've discussed, you need a way to unify that observability into a holistic view, normalize it, and consume that data.

Enhancing data-driven decisions with SLAs, SLIs, and SLOs

Considering key indicators such as SLAs, SLIs, and SLOs is an important part of putting an observability solution into play and using it to make data-driven decisions. Overall, the goal of an observability solution is to provide insight into the state of systems from data that can be collected and analyzed. By implementing these indicators in an observability solution, IT operation teams can better serve the companies they work with, helping them to reach their goals and better meet the needs and expectations of their clients.

An SLA describes what can be expected when consuming services or solutions from a service provider. These binding contracts establish real-world expectations for metrics such as network availability, security practices, defect rates, technical quality, and business results. They also define the repercussions for falling short of expectations.

[3] "The Race to IT Observability," LogicMonitor, 2021



Chapter 9: How close are you to full observability?

SLOs and SLIs break down SLAs into smaller pieces that can be measured on a technical level and are used by developer teams to gauge if they are truly meeting client expectations outlined within an SLA. All in all, SLIs form the basis of SLOs; and SLOs form the basis of SLAs.

SLAs, SLIs, and SLOs tie into observability because they establish guardrails for normal behavior. All defined metrics play roles in maintaining websites and other systems. The metrics get set because they match a company's needs. If performance falls below those standards, companies can expect challenges that interfere with their customer trust, branding, and reliability.

With a trustworthy observability suite, technology professionals can monitor their data, apps, and other assets while making sure third-party service providers continue giving them the resources they need to succeed. They can look at the system as a whole, concentrate on potential challenges, and monitor evolving needs as they experiment with new features, products, and customer services.

Assessing your state to observability

Understanding how close you are to full observability revolves around thinking about questions such as:

- How easy is it for you to obtain key telemetry such as logs, traces, and metrics?
- What level of analysis do you get from this telemetry — i.e. how useful is it to you?
- Do you have the ability to understand the internal states of your systems without significant additional coding and development?
- Can you gain a big-picture analysis of your whole system in real-time?

If your answers were generally positive, then you might be close to achieving full observability within your systems and software. Organizations that are far along in their observability journey are well-positioned to manage new changes and issues as they arise. Rapid scaling is rarely an issue because their detailed analysis will project this and offer solutions that can be easily implemented, without draining the existing resources of the system.

For organizations that have attained a high level of observability, problems with latency or infrastructure are easily identified thanks to effective traces combined with accurate logs and metrics, displayed effectively so they can be directly addressed. Downtime rarely happens, and when it does, it's for the minimal time possible because of the detailed and cohesive view and understanding of the systems involved. However, if you're still finding that you can't deal with these types of issues, it may be worth considering the tools or changes you need to make your system more resilient.

The next pillar in the ongoing strategy is to put end-to-end testing in place to measure the success of your initiative and ensure that the business processes are improving over time. Putting in the time up-front to carefully document your desired business outcomes and their relationships to your technology and processes will make it much easier to track your progress, and determine whether adjustments need to be made to get the most out of your investment in your observability initiative.

Steps to drive observability

There are a number of actions IT professionals can take in order to achieve unified observability across their company's applications, networks, and infrastructure. An internal observability orchestrator can serve as catalysts within your organization and know how to bring together the technical and business insights required to achieve full-stack observability. If you're seeking to champion observability within your organization, some key actions can help you move forward:

Embrace IT convergence and use it to your advantage

IT operations and administrative teams, as well as developers, have much to gain from breaking down silos and working together to uncover hidden insights and drive business results. Facilitating this convergence within their organization can help you ease the path to observability and enable your organization to better align processes and technology, with teams that work in lockstep to achieve your goals.

Prioritize and invest in a unified observability platform

Siloed data means siloed insights, so it's imperative to consolidate monitoring products and invest in a unified observability platform. Deploy a single platform that can collect, correlate, and put into context their applications, IT infrastructure, and log data. You should also implement AI and machine learning to diagnose issues before they become business problems. AI and ML not only empower your organization with more proactive responses but provide helpful insights to drive modernization.

Stay in sync with key IT trends and encourage your teams to adopt an agile approach to their work

The old saying that the only constant is change rang especially true in 2020. Shifting economic conditions, changing competitive landscapes, and unexpected challenges can test even the best-planned strategies. Encourage your IT teams to keep their eyes on the latest developments in technology, while also maintaining an agile mindset in order to adopt them quickly. Proactive, informed organizations will fare far better in the digital transformation race than competitors that lag behind.

Avoid business-disrupting outages at all costs

Downtime can undermine all the hard work you've invested in your strategic initiative. It's the one thing that negates every benefit of observability and overshadows every innovative IT trend. In today's hyper-competitive environment, businesses simply can't afford to experience outages and brownouts. Even a brief outage of a critical system or service can bring on heavy costs that extend far beyond the bottom line, and can impact the customer experience, brand reputation, and compliance.

Achieving observability may seem like a daunting task, but it's definitely in reach for organizations with the right strategy and a proven technology partner. IT leaders who follow the recommendations above will be well on their way to gaining the level of observability they need to make their organization more resilient and innovative. Ultimately, they will also help advance the important work of digital transformation, and provide a springboard for realizing enterprise business success.

©LogicMonitor 2021

Conclusion

Technologies and environments are continuing to progress at rapid speed, and to help their organizations compete successfully, IT professionals have no choice but to think and execute more strategically. Cloud adoption in particular is driving massive change. Gartner estimates that over 85% of organizations will embrace the cloud-first principle by 2025, with over 95% of new workloads being deployed on cloud-native platforms (up from 30% in 2021).[1]

Improving observability can help organizations tame the escalating complexities of IT. It goes beyond the limitations of traditional monitoring solutions, to empower IT with the context and correlation capabilities they need to make smarter, faster decisions. By bringing together the best capabilities of monitoring, machine learning, and log analysis, observability enables ITOps and DevOps professionals to more easily spot anomalies, uncover issues more proactively, and scale as their organization evolves.

Gartner predicts 30% of companies with cloud-based architectures will be employing observability techniques by 2024.

Observability is intended to unlock positive outcomes all across an organization, so it depends on a holistic approach, together with intelligent tools, to monitor, analyze, and trace events. Obtaining detailed metrics, traces, and logs delivers the fundamental telemetry data, but the value of observability goes deeper. After bringing together and analyzing the right data through monitoring and logs, observability lets organizations correlate the right pieces together, to better understand relationships and determine what to do with all the information they've gathered.

Achieving a holistic perspective is becoming easier, thanks to standards like OpenTelemetry that help teams instrument applications to generate high fidelity logs, metrics, and traces for their tools using an open format.

This holistic perspective is what enables observability to help developers and IT professionals, to make smarter decisions and drive outcomes that benefit the entire business, including:

- **Cost control and better resource utilization**
- **Faster innovation and improved business agility**
- **A common view of shared systems to help improve strategy**
- **Maximum availability of key business services**

Embrace IT convergence and use it to your advantage

When implementing observability, it's important not to be distracted by technology, but keep an eye on driving better outcomes. A successful observability strategy will be one that will fully align and scale to stay in sync with an organization's business strategy.

As they implement observability, organizations should examine their own processes and data, and align the tools, technical capabilities, and processes for observability to support them in a future-proof way. Encouraging a culture of observability is just as critical as the tools and platform that support it. ITOps and DevOps professionals should budget time for building telemetry into their services and deliverables, for smooth integration into the observability platform they choose.

With the right tools and technologies, backed by a company-wide observability mindset, organizations can unlock the full potential of complex, multi-cloud environments, and sharpen their competitive edge for the future.

[1] https://www.gartner.com/en/newsroom/press-releases/2021-11-10-gartner-says-cloud-will-be-the-centerpiece-of-new-digital-experiences

©LogicMonitor 2021

About LogicMonitor®

Monitoring unlocks new pathways to growth. At LogicMonitor®, we expand what's possible for businesses by advancing the technology behind them. LogicMonitor seamlessly monitors infrastructures, empowering companies to focus less on problem solving and more on evolution. We help customers turn on a complete view in minutes, turn the dial from optimization to innovation and turn the corner from sight to vision. Join us in shaping the information revolution by visiting LogicMonitor.com.

LogicMonitor