windward | dynatrace

PARTNER RESOURCE

Extending the pillars needed to achieve advanced observability.

Overview

In software, observability refers to the collection of measurements, better known as telemetry, produced by application services. Observability has been historically defined by three key pillars — metrics, distributed traces, and logs — the so-called “three pillars of observability”. Thanks to projects such as OpenTelemetry, which promotes the standardization of data collection, and W3C trace-context, built-in telemetry will soon become a must-have feature of cloud-native software.

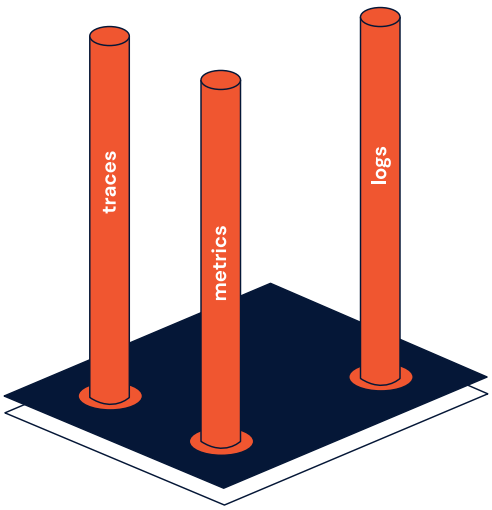
Dynatrace believes that metrics, traces, and logs are the beginning of achieving true observability, but there are several critical elements missing such as user experience and topology context. Observability needs to be considered with the end goal in mind; providing the full-stack context required to deliver causation-based, precise answers continuously and automatically. We call this Advanced Observability.

The three pillars of observability — Captured continuously and automatically, no code changes required

With Dynatrace, you just install a single agent — appropriately named OneAgent — per each monitored host to collect all relevant data, in context, with high fidelity. Dynatrace OneAgent auto-discovers all the processes running on a host, including dynamic microservices running inside containers. Based on what it finds, OneAgent automatically activates instrumentation specifically for your stack. New components are continuously auto instrumented in real-time, with no code changes or manual configuration required. This means that observability is automatically added to applications and telemetry data is available in real-time.

Sonja is a Technical Product Manager at Dynatrace. She’s responsible for the OneAgent SDK, distributed tracing, and open observability use cases. She also drives the topic of performance engineering.

You can follow her on Twitter: @SonjaChevre

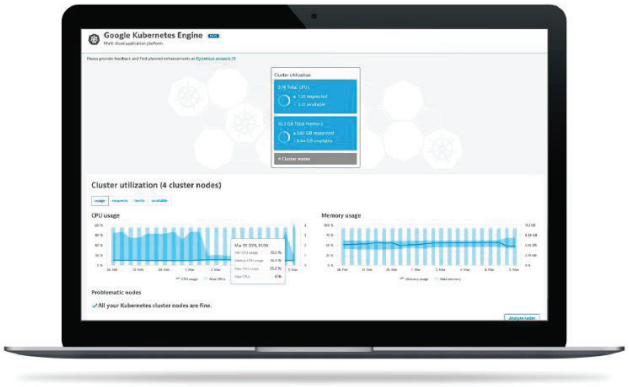


The three traditional pillars of observability

Metrics

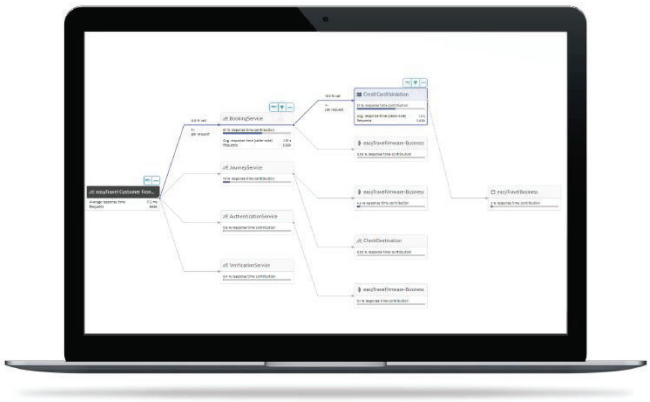
As a full-stack monitoring platform, Dynatrace collects a vast number of metrics for each OneAgent-monitored host in your environment. Depending on the types of technologies running on your hosts, the average number of metrics is about 500 per computational node.

Besides all the metrics that originate from your hosts, Dynatrace also collects all the important key performance metrics for services and real-user monitored applications, as well as metrics from cloud platforms like AWS, Azure, Kubernetes, and VMware Tanzu (formerly Pivotal). All told, thousands of metrics can be available to our Ai engine, Davis, and can also be charted, analyzed, and used for alerting Dynatrace.



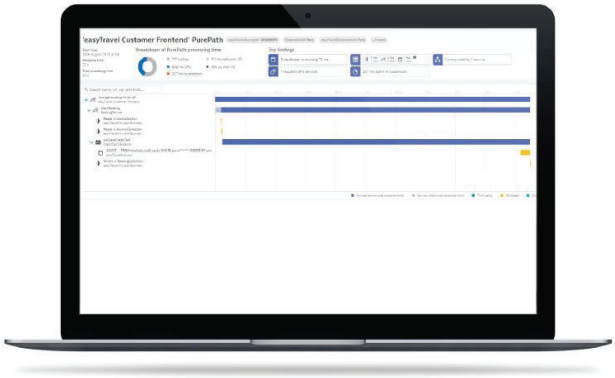
Distributed traces

Within a microservices environment, you can have dozens, if not hundreds, of services calling one another. Distributed tracing is used to understand how all those different services connect together, and how your requests flow through them. The Dynatrace service flows are automatically generated from distributed traces.

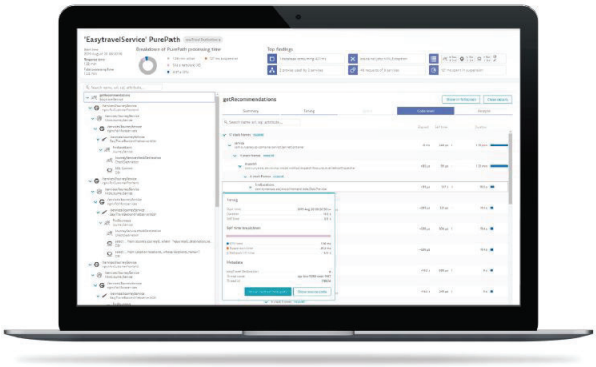


windward | dynatrace

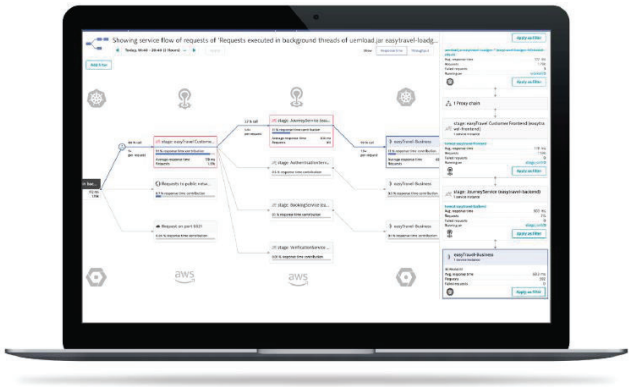
Distributed tracing, with no code changes required, has been a core component of Dynatrace from the very beginning. Our patented version of distributed trace, Dynatrace PurePath™, does even more as we not only capture tracing information but also code-level data. Distributed traces are extended with code level execution trees that include useful information such as CPU time, suspension time, and network I/O time.



Distributed tracing helps to rapidly (in our case, almost instantly) localize the service that's causing the issue. But distributed tracing won't help you dig deeper into the code to get more details. For that, you need OneAgent's code-level visibility and CPU profiling. OneAgent also captures information about background and service activity for CPU analysis and method hotspots. This is all done with OneAgent auto-instrumentation — which means no wasted time spent instrumenting or configuring.



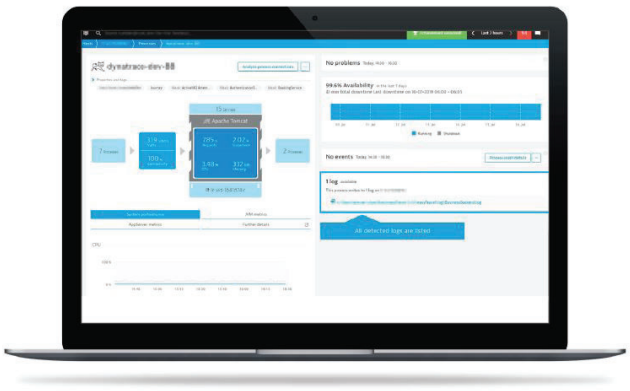
As the popularity of distributed tracing grows, it makes sense for the industry and community to standardize some of its aspects. So we're also leading the effort in the W3C distributed tracing working group.



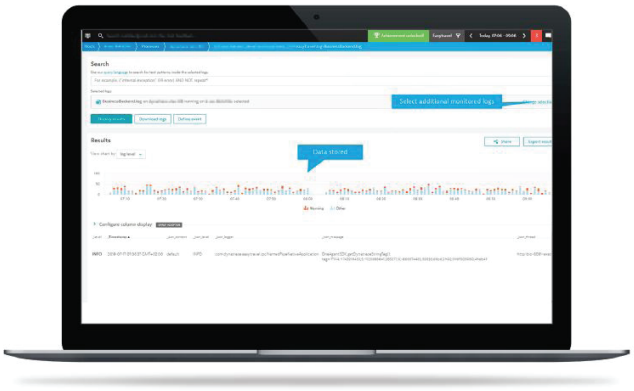
windward | dynatrace

Logs

OneAgent automatically detects log files and puts them in context with the corresponding host or process with no manual configuration. All detected log files are listed on the corresponding process group/process or host overview pages. For details, see log detection and supported log formats.



You can search for text patterns across multiple monitored logs and for alerts that are based on occurrences of those text patterns.



The fourth pillar: Front-end monitoring from the real-user perspective

While people tend to focus on back-end services when referring to observability, there is great value in starting traces on the client side (in the browser, mobile app or any custom application), pulling in metrics relating to user experience, and extracting user data from logs.

Dynatrace Digital Experience Monitoring (DEM) automatically captures full sessions from real users, providing you with complete visibility into the customer experience across every digital touchpoint. Without this “User Experience” pillar added into the definition of advanced observability, you’re missing context relating to HTTP and Javascript errors, application performance from the user’s perspective, and third-party services. You would also have blind spots to the impact that applications have on business-level KPI’s such as revenue, conversions, and customer engagement.

If you define advanced observability as a means to the goal of delivering precise answers thar are actionable and prescriptive in what you need to do, then understanding the impact to the user and business are essential to make the right decisions.

Avoid data silos: Context matters

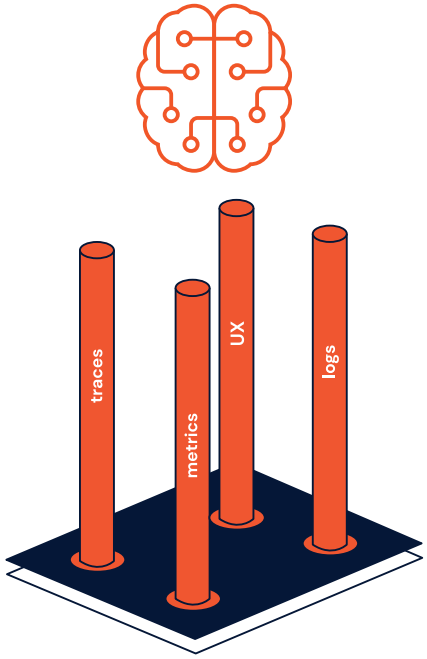
Capturing observability data is good. But it's even better when it's done automatically, without code changes or cumbersome manual configuration needed from your already constrained human resources, as OneAgent does.

However, too often, metrics, traces, logs, and user experience are treated as data silos, with no relationships between them and are aggregated in meaningless ways without any answers. But what is the point of capturing and reporting on this disconnected data? Looking at metrics, traces, logs, and user experience alone — without meaningful interdependencies and context between the pillars — is useless for root cause analysis in highly dynamic and complex environments.

With silos, you might get an alert for an increase in the failure rate of service A. And then get an alert because process B has an increase in CPU usage. But you won't know if these two alerts have a causational relationship and how your users may be impacted by them. You need to dig deeper, but you're looking for only a few traces amongst billions that document the issue. It's going to be extremely difficult to find that needle in the haystack. And it will be near impossible to do it continuously as your environment continues to scale in size and complexity.

The secret to connecting the silos with context?

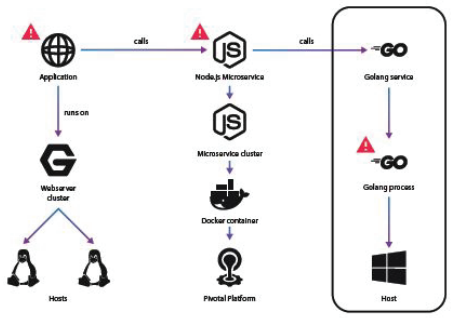
You need to take an expanded approach to observability with real-time topology discovery and dependency mapping in order to get precise answers. With Smartscape™, Dynatrace continuously and automatically maps data into a real-time dependency map that shows the relationships and dependencies for all entities, both virtually up and down the stack and horizontally between services, processes, and hosts. All captured traces, metrics, log files, and user experience data points are automatically mapped to the monitored entities they relate to.



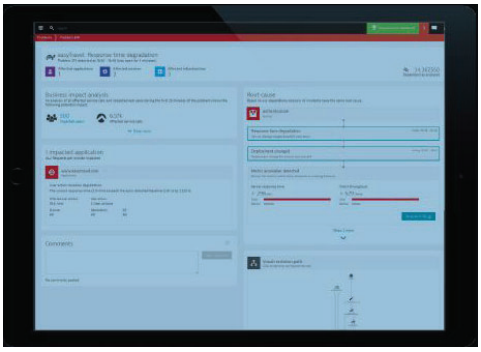
Advanced observability

windward | dynatrace

By automatically analyzing the dependencies among components, Dynatrace identifies not only if a problematic service is the root cause of a problem, but also its dependencies on other services that run within different process groups in your data center. In the example to the left, Dynatrace's AI engine, Davis, automatically follows and evaluates the health of each relevant back-end service within the transaction and immediately identifies a Golang service as the root cause.



As a result, Dynatrace AI presents the complete, vertical technology stack of the Golang service in the Root cause section of the problem details page and highlights all the relevant findings. With one click, you can dive deep into the affected metrics, traces, log files, and user experience to analyze and fix the issue.



Having advanced observability into what you know could go wrong can make a huge difference. But what can really ruin your business are the unknown unknowns. These are the things you aren't aware of, don't understand, and don't monitor. Because of how Dynatrace is architected, it automatically monitors unknown unknowns in your environment. Our platform auto-instruments, captures high-fidelity data, and provides full-stack real-time dependency maps to enable instant, precise, AI-assisted answers—not just correlations. By extending the pillars and requirements of advanced observability, you can finally achieve precise answers that drive action and significantly improved business outcomes.

LET'S TALK

Ready to find flow?

Windward helps companies create an IT operations strategy that connects your vision to a roadmap for success. If you'd like to learn more and discuss a strategic IT Ops plan for your organization, feel free to email us at info@windward.com or go to www.windward.com.