Site24x7

Understanding and Achieving Observability - A Primer

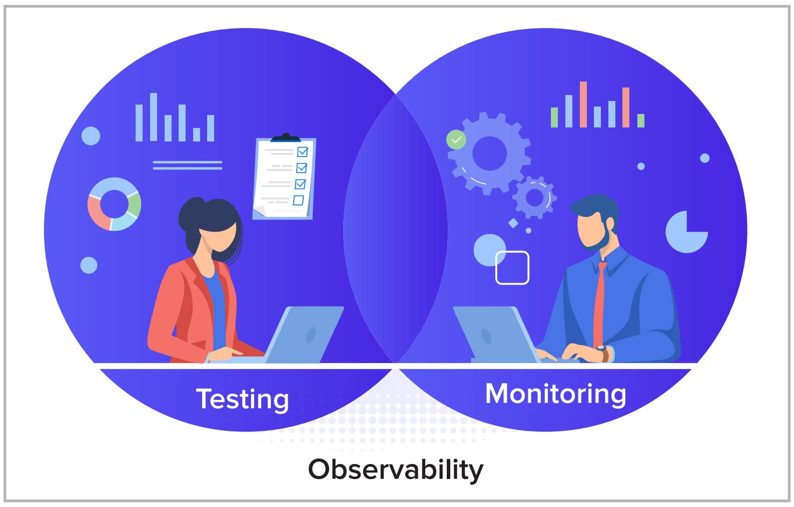
WHITEPAPER

Synopsis

Observability is the talk of the town in the DevOps community these days. While IT veterans argue what the definition of observability is, it's important to note that observability does not exclude monitoring. In fact, observability is a superset of testing and monitoring enabling you to have a holistic view of your application's behavior and health at any given point of time. This white paper aims to discuss the definition of observability, its evolution, what it means from person to person, and how to achieve it.

Introduction

As the role of DevOps evolves from being reactive to operational issues, to being able to proactively identify issues, it's critical to prepare systems for effective monitoring and correlate data from all available sources. Here we take a look at yet another evolving trend on making systems observable.



What is observability?

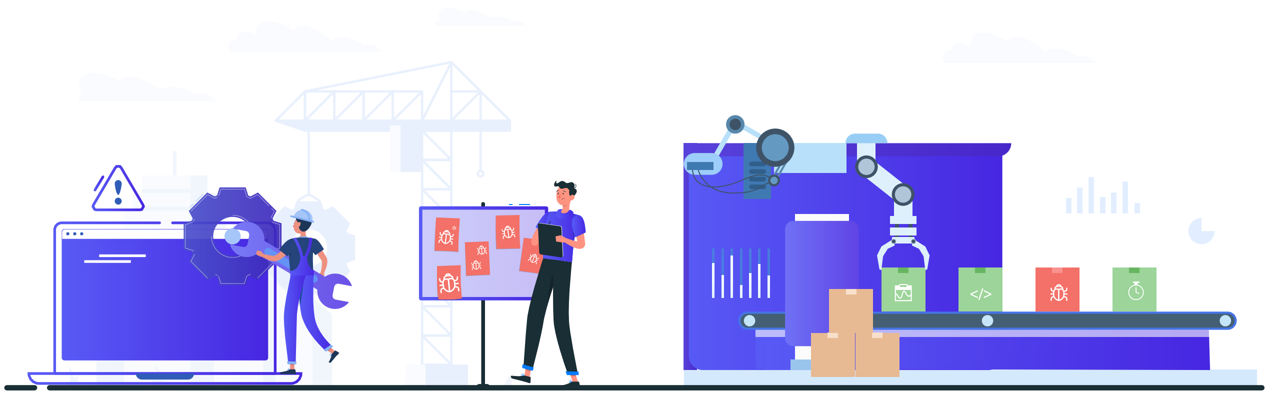
It's hard to define an evolving buzzword as each segment of the industry associates differently to it. For the purpose of this article, we define observability as:

“A state of readiness of the DevOps team to instantaneously identify unexpected application behavior and pinpoint its root cause.”

Shift-left Testing and Observability

The utopian dream of any DevOps team is to have no unknown issues in their production environment. When an issue is identified, DevOps teams try to understand why it happened and include it as a test case in the testing process. This is commonly known as shift-left testing. The shift-left testing is important for DevOps to stay ahead in the competitive landscape.

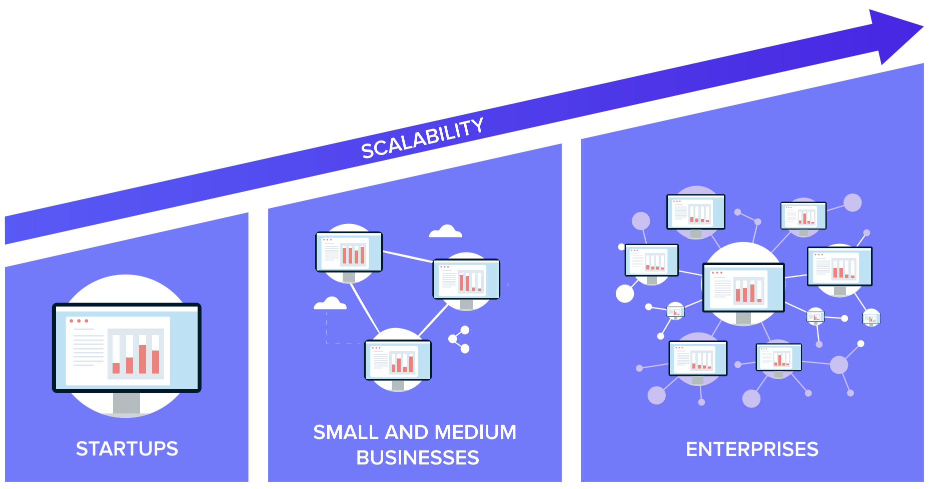
With the growth in monitoring and automation tools, DevOps teams are able to "shift-left" their quality assurance (QA) and testing process. By making systems more observable, DevOps can easily understand how to quickly shift-left any new problems.



Setting Goals for Observability

The monitoring needs of an application/service largely depend on its scale and size. A small-scale application may not need much monitoring, but as the scale and complexity of applications/services grows, so does its monitoring needs. This is why the scope of monitoring and observability depends on your organization's size and the scale of its services.

For instance, a web hosting provider may only need website and infrastructure monitoring to notify its customers when its service becomes unresponsive, while the customers themselves may need log and the application performance monitoring to keep up with their SLAs. On the other hand, a more sophisticated payment gateway service that handles millions of transactions a day would need continuous monitoring of all transactions, logs, and business metrics, with smart alerting and response systems that can perform automated actions during failures and generate reports on the root cause analysis for outages.



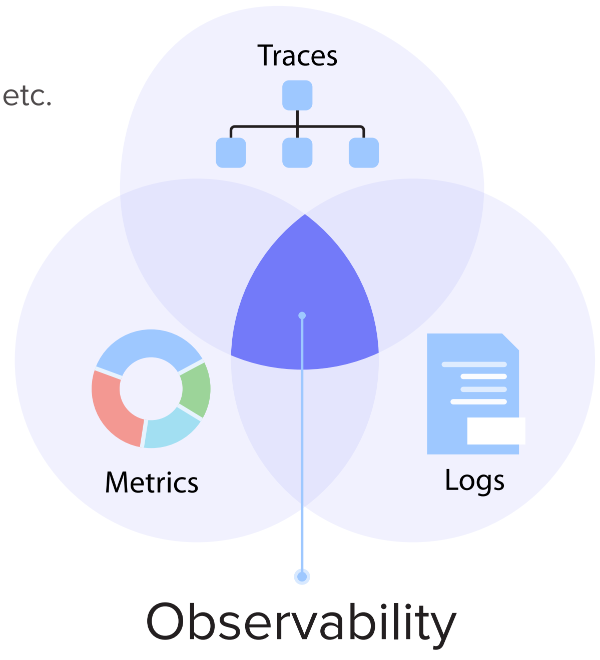
Uptime Monitoring, Application Monitoring, Infrastructure Monitoring, Cloud Monitoring

Monitoring goals vary according to an organization's specific needs, which means the goal of observability depends on what you want to monitor; it may vary from person to person within your organization, and will differ from other organizations' goals.

Achieving Observability: Pillars and Pitfalls

Once you've defined your goals of monitoring and observability, you have to focus on refining your application life cycle process to achieve these goals. The three main areas of focus, commonly known as the three pillars of observability, are metrics, logs, and traces.

1. Metrics: There are industry standard metrics like Apdex, average response time, 95th percentile lines, etc. to get you started. But you can go further by adding business-specific metrics like scheduled jobs, third-party resource utilizations, key component usage, etc.



2. Logs: Application and server logs can provide request-specific details that cannot be captured via standardized monitoring solutions. While app server logs provide request-specific details, such as user agent, features accessed by the users, and IP addresses, developers can add contextual information about the state of the system. Analyzing the logs tells the story of system behavior over time.

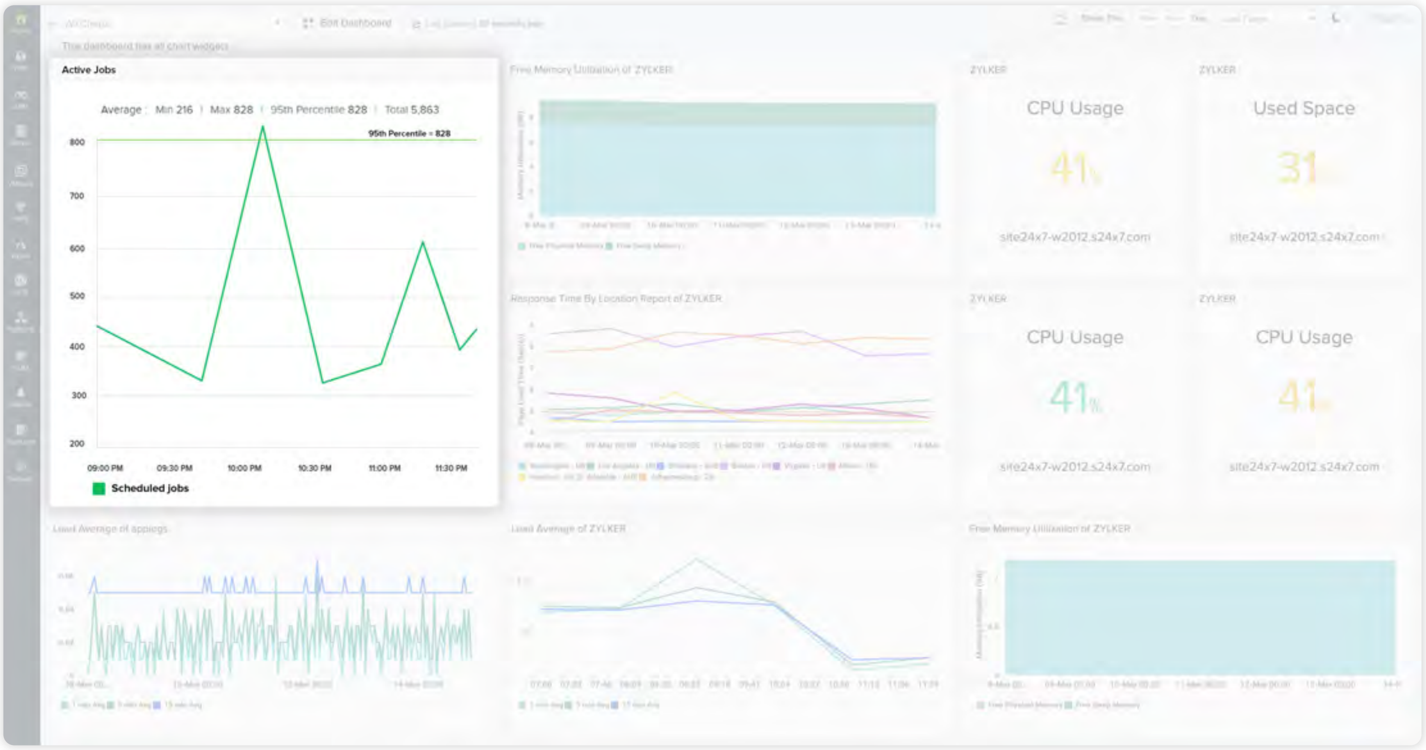
3. Traces: Debugging slow transactions requires insights into time-consuming methods and external services. Traces provide a snapshot of the entire workflow of a transaction, making it easier to identify parts of the code that require optimization and fine-tuning.

Achieving Observability: Guidelines

Once the goals of observability are established, it's time to align your monitoring solutions with them. Following these guidelines, you'll be able to get the most out of your monitoring solution and tailor-fit it for your needs.

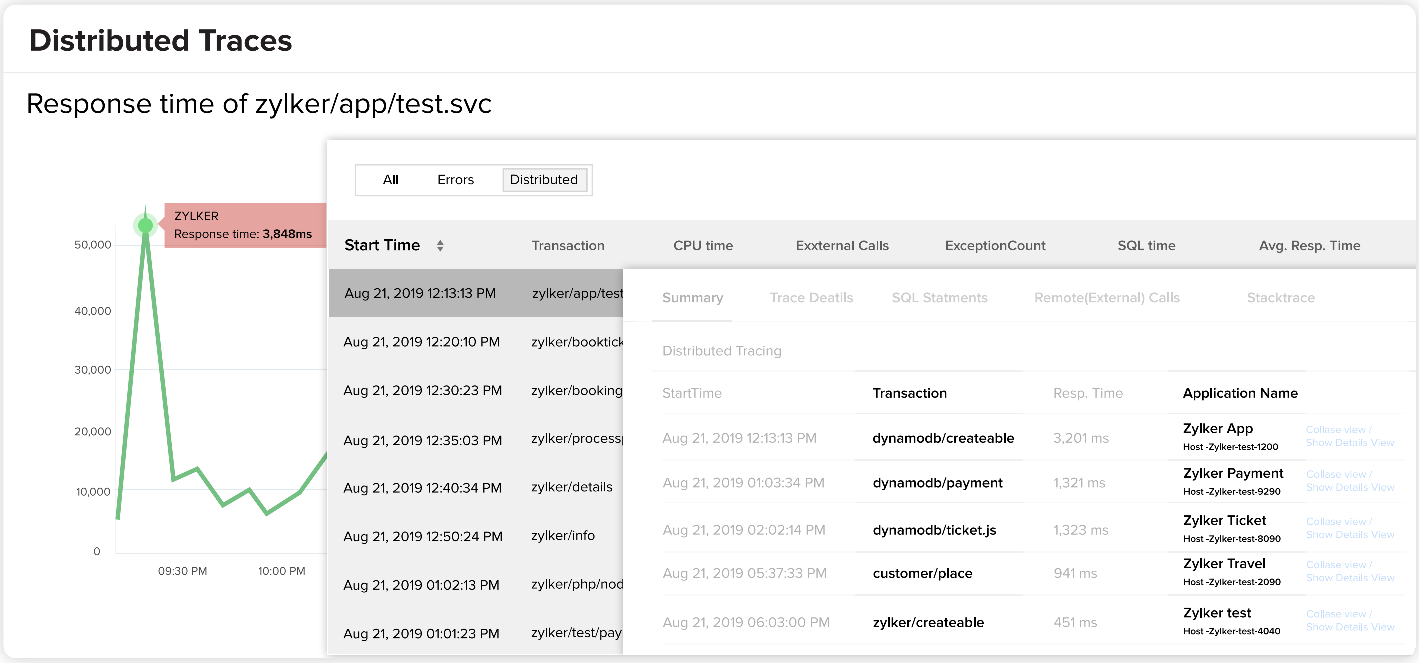
1. Use custom metrics: Monitoring tools have done a great job to get you started in troubleshooting by capturing numerous industry standard metrics. However, they can only provide you an outsider's perspective on your service. Defining and monitoring application-specific metrics may increase the development workload, but it's definitely worth the trouble since it can provide deep context into the different states of your system to help you understand what factors need to be tuned to achieve optimal performance.

> Site24x7 manages a lot of scheduled jobs to run customer-specific workloads. By monitoring the active jobs, we can determine how to scale the servers and improve resource utilization.Since the jobs are running as separate threads spawned across multiple servers, monitoring them is only possible by defining custom metrics.



2. Analyze traces: Monitoring slow application programming interfaces (APIs) is a great way to increase observability of a system. Dig deep through traces to discover bottlenecks in the performance of your application/service. If a third-party is slowing you down, find ways to replace or avoid the third-party service. If the database is slowing you down, consider upgrading or de-normalizing your database. If you have a distributed system, use a monitoring solution that can provide distributed traces. With distributed traces, you can not only optimize individual applications, but rework the communication flows and enhance the entire network.

Site24x7 offers distributed tracing, allowing you to monitor code flows across application boundaries.



Distributed Traces

Response time of zylker/app/test.svc

Start Time	Transaction	CPU time	Exxternal Calls	ExceptionCount	SQL time	Avg. Resp. Time
Aug 21, 2019 12:13:13 PM	zylker/app/test					
Aug 21, 2019 12:20:10 PM	zylker/booktick					
Aug 21, 2019 12:30:23 PM	zylker/booking					
Aug 21, 2019 12:35:03 PM	zylker/process					
Aug 21, 2019 12:40:34 PM	zylker/details					
Aug 21, 2019 12:50:24 PM	zylker/info					
Aug 21, 2019 01:02:13 PM	zylker/php/nod					
Aug 21, 2019 01:01:23 PM	zylker/test/pay					

Distributed Tracing

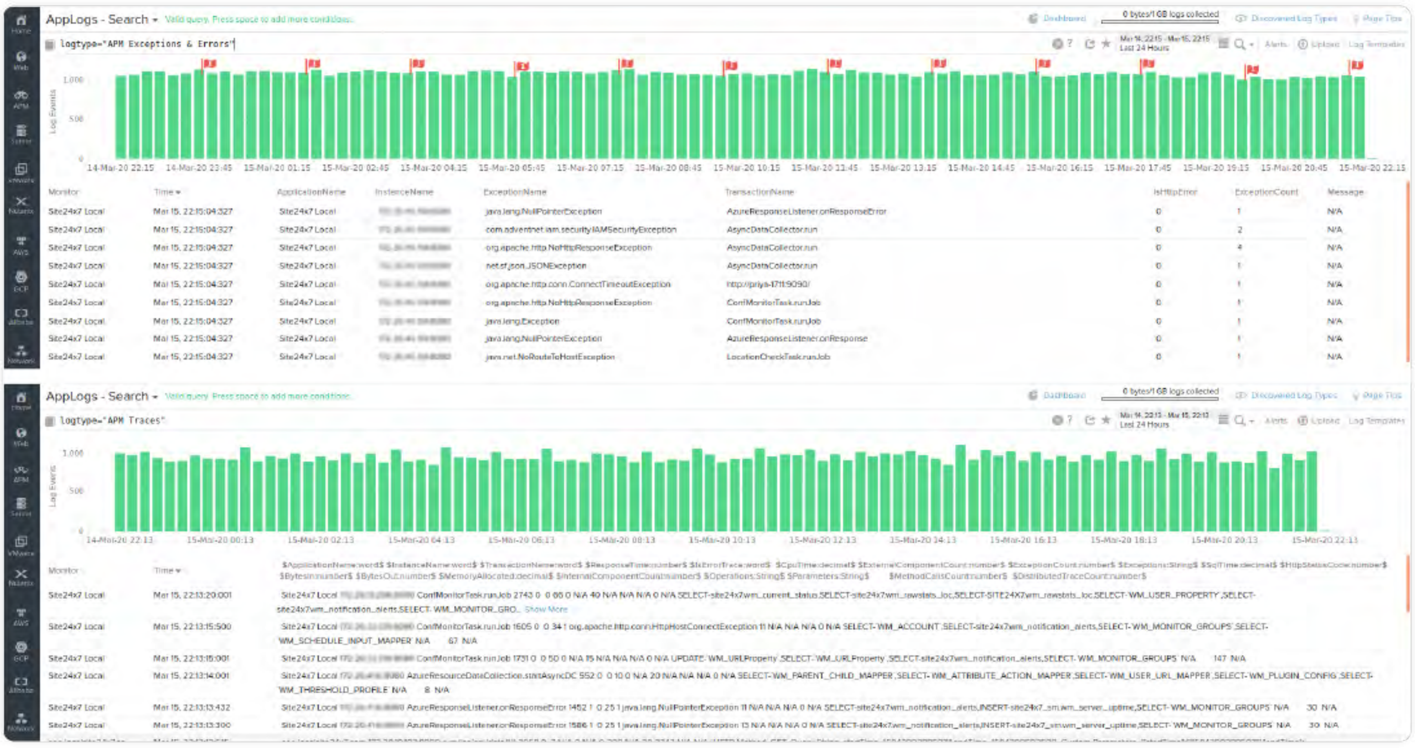
StartTime	Transaction	Resp. Time	Application Name	
Aug 21, 2019 12:13:13 PM	dynamodb/createable	3,201 ms	Zylker App Host -Zylker-test-1200	Collase view / Show Details View
Aug 21, 2019 01:03:34 PM	dynamodb/payment	1,321 ms	Zylker Payment Host -Zylker-test-9290	Collase view / Show Details View
Aug 21, 2019 02:02:14 PM	dynamodb/ticket.js	1,323 ms	Zylker Ticket Host -Zylker-test-8090	Collase view / Show Details View
Aug 21, 2019 05:37:33 PM	customer/place	941 ms	Zylker Travel Host -Zylker-test-2090	Collase view / Show Details View
Aug 21, 2019 06:03:00 PM	zylker/createable	451 ms	Zylker test Host -Zylker-test-4040	Collase view / Show Details View

3. Structure logs for data mining: While metrics and traces alone can give a holistic picture of your application's day-to-day operational behavior, they are limited to monitoring internal factors. Your application performance may be affected by external factors like user behaviors, which cannot be captured by metrics and traces.

This is where logging can provide context. Is a single rogue agent choking your bandwidth? Is application performance only taking a hit from certain geographies and browsers? Is the database constantly bombarded with the same query, the results of which can be cached? These questions can only be answered by analyzing logs. Log monitoring makes answering these questions a breeze. However, without the context of traces and custom metrics, the answers to these questions can very well lead your performance optimizations the wrong way.

If you structure your logs so that they can be queried, associating them with traces and other metrics becomes more natural and relevant. Restructure your logs to include custom metrics and use traces to identify where logging is more relevant. Once you bring the context of custom metrics and traces into your logs, you will have a holistic view of your application's state. You can then feed this data to tools that can perform further analysis.

> In Site24x7, all service logs are pushed to our central log server from which we query and monitor infrastructure events. Site24x7's Log Management console provides querying functionality along with trend analysis, which makes it easier to correlate and arrive at root cause analysis (RCA).



```
logtype="APM Exceptions & Errors"
```

Monitor	Time	ApplicationName	InstanceName	ExceptionName	TransactionName	IsHttpError	ExceptionCount	Message
Site24x7 Local	Mar 15, 22:15:04:327	Site24x7 Local	[redacted]	java.lang.NullPointerException	AzureResponseListener.onResponseError	0	1	N/A
Site24x7 Local	Mar 15, 22:15:04:327	Site24x7 Local	[redacted]	com.adventnet.iam.security.IAMSecurityException	AsyncDataCollector.run	0	2	N/A
Site24x7 Local	Mar 15, 22:15:04:327	Site24x7 Local	[redacted]	org.apache.http.NoHttpResponseException	AsyncDataCollector.run	0	4	N/A
Site24x7 Local	Mar 15, 22:15:04:327	Site24x7 Local	[redacted]	net.sf.json.JSONException	AsyncDataCollector.run	0	1	N/A
Site24x7 Local	Mar 15, 22:15:04:327	Site24x7 Local	[redacted]	org.apache.http.conn.ConnectTimeoutException	http://priya-1711:9090/	0	1	N/A
Site24x7 Local	Mar 15, 22:15:04:327	Site24x7 Local	[redacted]	org.apache.http.NoHttpResponseException	ConfMonitorTask.runJob	0	1	N/A
Site24x7 Local	Mar 15, 22:15:04:327	Site24x7 Local	[redacted]	java.lang.Exception	ConfMonitorTask.runJob	0	1	N/A
Site24x7 Local	Mar 15, 22:15:04:327	Site24x7 Local	[redacted]	java.lang.NullPointerException	AzureResponseListener.onResponse	0	1	N/A
Site24x7 Local	Mar 15, 22:15:04:327	Site24x7 Local	[redacted]	java.net.NoRouteToHostException	LocationCheckTask.runJob	0	1	N/A

```
logtype="APM Traces"
```

```
Monitor          Time                   $ApplicationName:word$ $InstanceName:word$ $TransactionName:word$ $ResponseTime:number$ $IsErrorTrace:word$ $CpuTime:decimal$ $ExternalComponentCount:number$ $ExceptionCount:number$ $Exceptions:String$ $SqlTime:decimal$ $HttpStatusCode:number$
                                        $BytesIn:number$ $BytesOut:number$ $MemoryAllocated:decimal$ $InternalComponentCount:number$ $Operations:String$ $Parameters:String$      $MethodCallsCount:number$ $DistributedTraceCount:number$
Site24x7 Local   Mar 15, 22:13:20:001   Site24x7 Local [redacted] ConfMonitorTask.runJob 2743 0  0 66 0 N/A 40 N/A N/A N/A 0 N/A SELECT-site24x7wm_current_status,SELECT-site24x7wm_rawstats_loc,SELECT-SITE24X7wm_rawstats_loc,SELECT-'WM_USER_PROPERTY',SELECT-
                                        site24x7wm_notification_alerts,SELECT- WM_MONITOR_GRO... Show More
Site24x7 Local   Mar 15, 22:13:15:500   Site24x7 Local [redacted] ConfMonitorTask.runJob 1605 0  0 34 1 org.apache.http.conn.HttpHostConnectException 11 N/A N/A N/A 0 N/A SELECT-'WM_ACCOUNT',SELECT-site24x7wm_notification_alerts,SELECT-'WM_MONITOR_GROUPS',SELECT-
                                        'WM_SCHEDULE_INPUT_MAPPER' N/A      67  N/A
Site24x7 Local   Mar 15, 22:13:15:001   Site24x7 Local [redacted] ConfMonitorTask.runJob 1731 0  0 50 0 N/A 15 N/A N/A N/A 0 N/A UPDATE-'WM_URLProperty',SELECT-'WM_URLProperty',SELECT-site24x7wm_notification_alerts,SELECT-'WM_MONITOR_GROUPS' N/A      147  N/A
Site24x7 Local   Mar 15, 22:13:14:001   Site24x7 Local [redacted] AzureResourceDataCollection.startAsyncDC 552 0  0 10 0 N/A 20 N/A N/A N/A 0 N/A SELECT-'WM_PARENT_CHILD_MAPPER',SELECT-'WM_ATTRIBUTE_ACTION_MAPPER',SELECT-'WM_USER_URL_MAPPER',SELECT-'WM_PLUGIN_CONFIG',SELECT-
                                        'WM_THRESHOLD_PROFILE' N/A      8  N/A
Site24x7 Local   Mar 15, 22:13:13:432   Site24x7 Local [redacted] AzureResponseListener.onResponseError 1452 1  0 25 1 java.lang.NullPointerException 11 N/A N/A N/A 0 N/A SELECT-site24x7wm_notification_alerts,INSERT-site24x7_sm.wm_server_uptime,SELECT-'WM_MONITOR_GROUPS' N/A      30  N/A
Site24x7 Local   Mar 15, 22:13:13:300   Site24x7 Local [redacted] AzureResponseListener.onResponseError 1586 1  0 25 1 java.lang.NullPointerException 13 N/A N/A N/A 0 N/A SELECT-site24x7wm_notification_alerts,INSERT-site24x7_sm.wm_server_uptime,SELECT-'WM_MONITOR_GROUPS' N/A      30  N/A
```

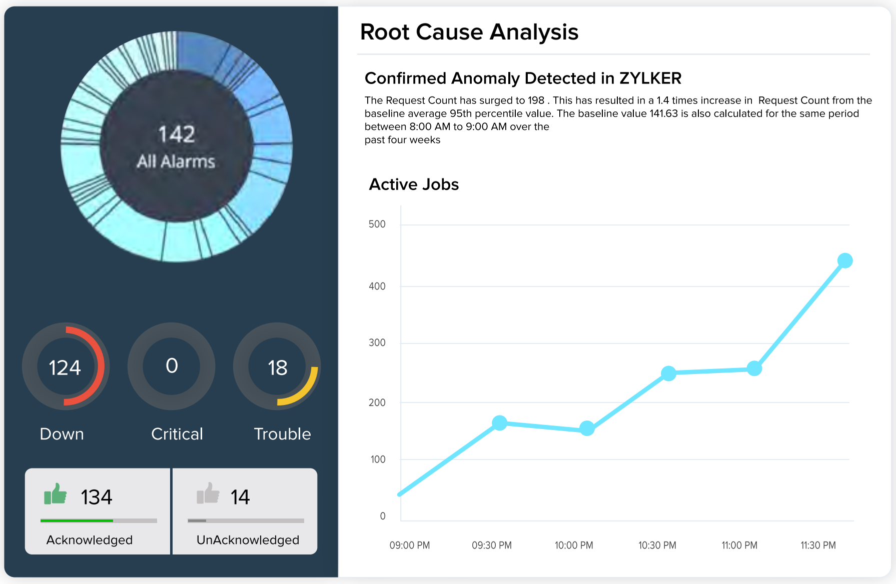
If you structure your logs so that they can be queried, associating them with traces and other metrics becomes more natural and relevant. Restructure your logs to include custom metrics and use traces to identify where logging is more relevant. Once you bring the context of custom metrics and traces into your logs, you will have a holistic view of your application's state. You can then feed this data to tools that can perform further analysis.

4. Perform trend analysis: Having a lot of structured data is only the starting point of performance tuning. You also need complete visibility on how each change affects the overall performance of your system. To see this, you need a solution that can store historic data and perform trend analysis, preferably with some ML/AI built-in to detect anomalies and alert you whenever the system does something unexpected. After gathering enough data-points and identifying the relations between them, you can even begin to automate some decisions.

5. Set up alerting and automation: An ideal alert system should be intelligent enough to distinguish patterns of application behavior under normal and extreme circumstances. Getting bombarded with alerts that end up getting ignored is just as good as having no alert system at al

For setting up intelligent alerts, many tools integrate ML/AI to study the application behavior over time and detect anomalies. However, the best alerting system can only be built by carefully configuring critical parameters based on experience. For instance, if your disks are getting full frequently, you can set up alerting when disk utilization reaches 80 percent, and set up automated scripts to run when thresholds are breached, like archiving old data or clearing up old logs, cached files, etc. Understanding the system well enough to make these informed decisions comes from experience. However, with reports from monitoring solutions, these decisions can be made much more quickly, like archiving old data or clearing up old logs, cached files, etc.

Site24x7 has automated actions to not only mitigate a failure, like retrying failed jobs, but also to collect diagnostic information during abnormal circumstances, like memory dumps when usage is high. These actions make the job of DevOps teams easier, since they can attend to only critical events, and let the automation handle the rest.



Confirmed Anomaly Detected in ZYLKER

The Request Count has surged to 198 . This has resulted in a 1.4 times increase in Request Count from the baseline average 95th percentile value. The baseline value 141.63 is also calculated for the same period between 8:00 AM to 9:00 AM over the past four weeks

Conclusion

You get to decide what level of observability you want in your system. Use your resources, instrument your metrics, and set up proactive alerting to correlate information across various silos to get the best information possible. In short, observability is neither a tool nor a mechanism, but more of a set of practices/culture that helps you to align your goals.

About the Author

Ranjani Subramanian works as a Product Marketer at Site24x7. She is committed to generate meaningful and informative content that educates the readers. Her areas of interest revolves around application performance monitoring, digital experience management and end-user experience. While not at work, you can always find her talking nonchalantly about movies and Netflix.

Ranjani PS
Product Marketer