Observability

100-level live demo script

Introduction

Actions

- From sidebar menu, click on **Applications**
- Choose **Robot Shop**, and click on the **Dependencies** tab

Narration

In this demo, I'll show how IBM Instana helps quickly identify, debug, and resolve an incident in a microservices-based application.

To set the context, our application is called Stan's Robot Shop, and it uses various technologies such as Java, Python, and MySQL.

This is a visualization of all the dependencies within the robot shop application. Instana automatically discovered the relationships between the services and correlated them into this dynamic graph. We can see how requests are moving through the application in real-time. Instana is able to do this because it tracks every request that flows through the application.

We can tell there are some problems with the application because several of the services are highlighted in yellow and red. But, you wouldn't normally be looking at the dashboard when something like this happens, so let me walk you through what it looks like from the SRE/IT operator's point of view when an incident occurs.

1 - Getting an incident alert

1.1 - Automatically assessing events and alerts

Action

- Click on the **Events** icon (triangle) in sidebar menu

Narration

We've just gotten an alert from Instana that there has been a sudden increase in erroneous calls on our 'discount' service, which is part of the robot shop application.

Although I don't have it connected right now, the alert would show up via one of the configurable alert channels, like email, Microsoft Teams, Slack, and many others (full list).

It's important to note here that you're not getting alerts for just anything. Behind the scenes, Instana is determining which events and issues are related, and it only sends alerts if a problem is likely to affect end users.

Let's go into the details for this incident.

2 - Inspect auto-correlated incident details

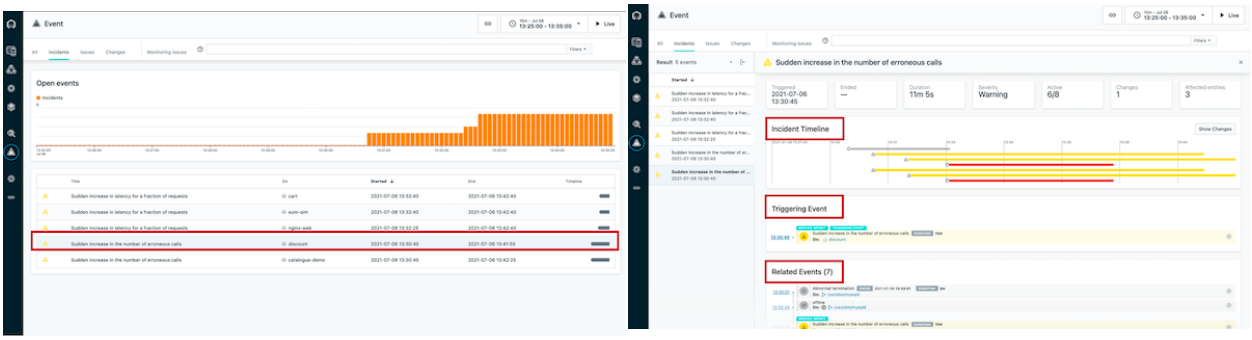
2.1 - Gather information from incident detail page

Action

- Click on the incident called **Sudden increase in the number of erroneous calls** on the ‘discount’ service

Narration

Instana recognized that the sudden increase in the number of erroneous calls was something important to alert on, so we did not have to do any configuration or set thresholds in order to get this alert. We get key information right away when we come into this incident detail page. There’s a timeline of the incident, the event that triggered Instana to create the incident, and all of the related events.



3 - Debugging the incident by inspecting calls

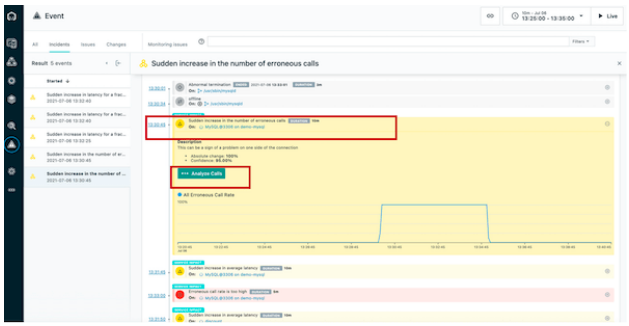
3.1 - Understand the incident

Actions

- Under **Related Events**, click on the event that says **Sudden increase in the number of erroneous calls**
- Then, click **Analyze Calls**

Narration

It looks like the abnormal termination of the MySQL database caused the problem. But we want to confirm this, and Instana makes it very easy to dig deeper into each individual call.



3.2 - Examine details

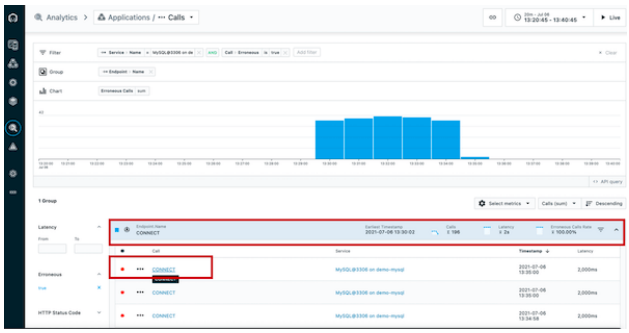
Actions

- Click on the endpoint named **CONNECT**
- Then, click on the first call (also named **CONNECT**)

Narration

All the calls are grouped by endpoint. There is only one endpoint here, but if there were multiple, you'd see a list here. Endpoints are automatically discovered and mapped by Instana.

We can go into the details for each erroneous call to MySQL via this endpoint (CONNECT).



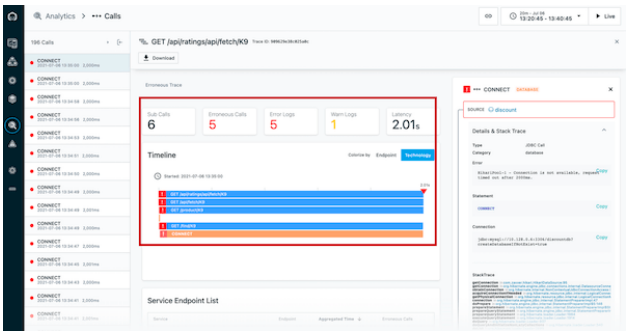
4 - Drill down with end-to-end traces

4.1 - View call via visual dashboard

Action
None

Narration
Clicking on an individual call takes us to a view of the call in context of the end-to-end trace. We can see where the request began and each call that was made along the way.

Everything is presented in an easy-to-navigate visual dashboard, so we can drill into increasingly detailed information to pinpoint the problem, without using multiple tools or navigating back and forth to lots of dashboards.



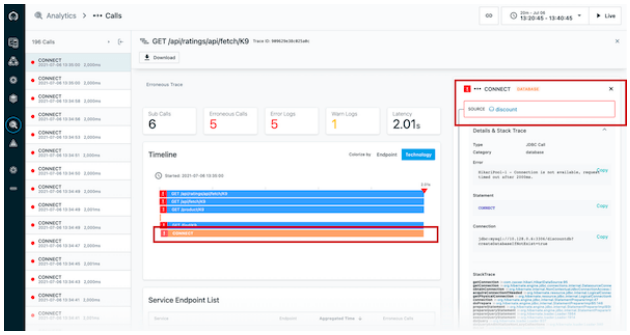
4.2 - Understand the impact and source of the incident

Action
- Click on the span called **CONNECT** and refer to the sidebar on the right side

Narration
Clicking on a span gives more details, including the source and destination as well as the full stack trace. In this case, we can see that the source is the ‘discount’ service, and [scroll down] the destination is CONNECT of MySQL.

So we can confirm that the root cause of the incident that affected the ‘discount’ service was with the MySQL database. The abnormal termination of the database caused a connection error, which then flowed back through the application.

When we bring MySQL back online, it will fix the problem.



5 - Confirm incident resolution was successful

5.1 - Metrics for the robot shop have returned to normal

Actions

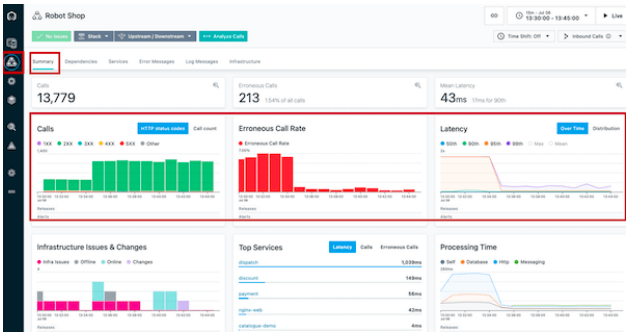
- Set the timeframe so it begins at 0:30 minutes past the hour and ends at 0:45 minutes past the hour
- Navigate to **Applications** in the sidebar menu, choose **Robot Shop**, and click on the **Summary** tab

Note

- You should see that the call volume has increased, the number of erroneous calls decreased, and latency also decreased

Narration

Now that MySQL is working again, we can go back and confirm that the problems with the robot shop have been repaired.



Summary

Now, we can see that the metrics for the robot shop have returned to normal: the call volume has increased again, the erroneous call rate has decreased, and latency has decreased.

We've fixed the problem with the robot shop and restored normal service!

Hopefully, you've seen that Instana can help make the process of identifying problems and finding the root cause of those problems very frictionless. Since Instana automates so many of the manual and labor-intensive aspects of the process, you can focus on getting other work done and not worry about instrumenting observability or constantly monitoring for problems. And when problems do arise, all the trace data is there at your fingertips to dig into.