NOTE: The provided demo environment uses the Instana SaaS service, which follows the bi-weekly release schedule. This may result in minor differences between the screenshots in this guide and the actual environment you will see, however, in most cases, it will not impact the demo flow. In case you encounter significant differences please report in Slack support channel #itz-techzone-support.
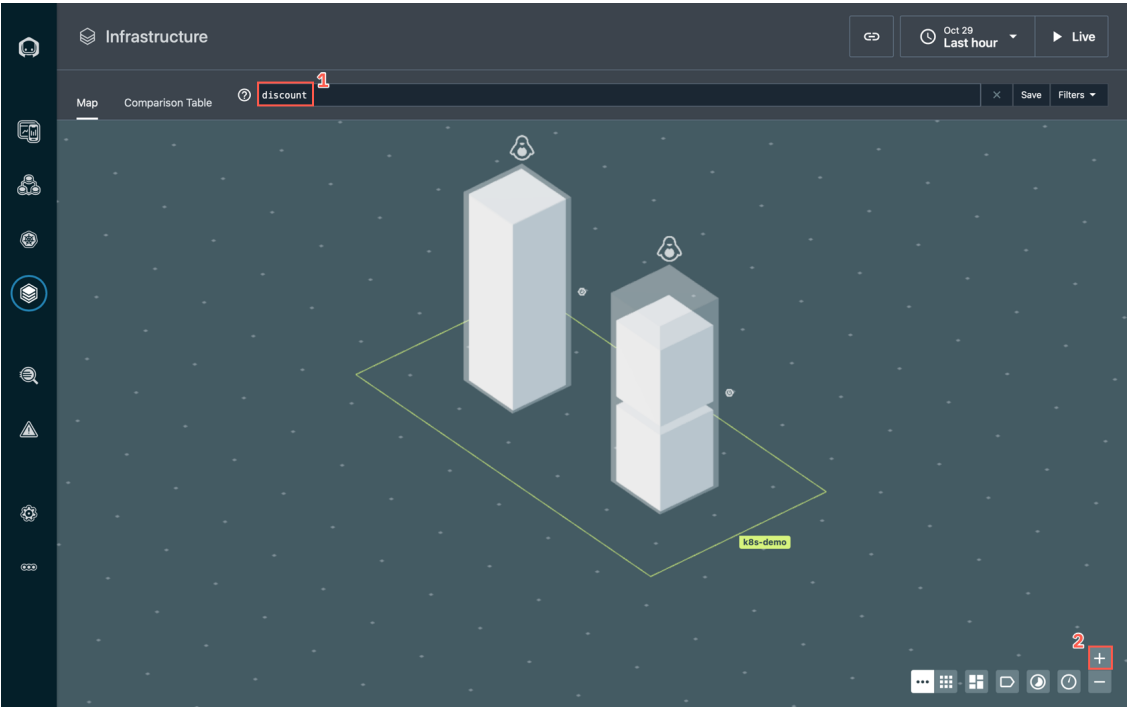
To access the demo environment:

- For IBMers, click here
- For business partners, click here

Introduction

Starting point

|	
Actions	From the sidebar menu, click **Infrastructure**. Choose **Robot Shop.**
Narration	In this demo, I will show you a quick overview of Instana's Enterprise Observability Platform. Instana automates observability, with a single agent per host, application components are discovered and observed as they are deployed. Over 200 technologies are supported with zero or minimal configuration, releasing you from installing and configuring multiple plugins. Instana's automation frees up valuable engineering time and ensures code is instrumented to a common standard.
Screenshot	

1. Infrastructure map

1.1 Exploring infrastructure map

Actions	Enter **discount** in the Search Bar (1). Zoom the view using the **'+'** sign in the bottom-right corner (2).
Narration	As well as metrics, logs and traces, Instana automatically indexes entity metadata such as: container labels, Kubernetes deployment labels and application framework names. This makes it easy to find any component, for example the discount service.

Screenshot

1.2 Drilling into the host

Actions	Click the light gray box representing the Linux host (1). Then, click **Open dashboard** button (2).
Narration	Drilling into this host shows its metadata in the left panel and the time series metrics on the right.

Screenshot
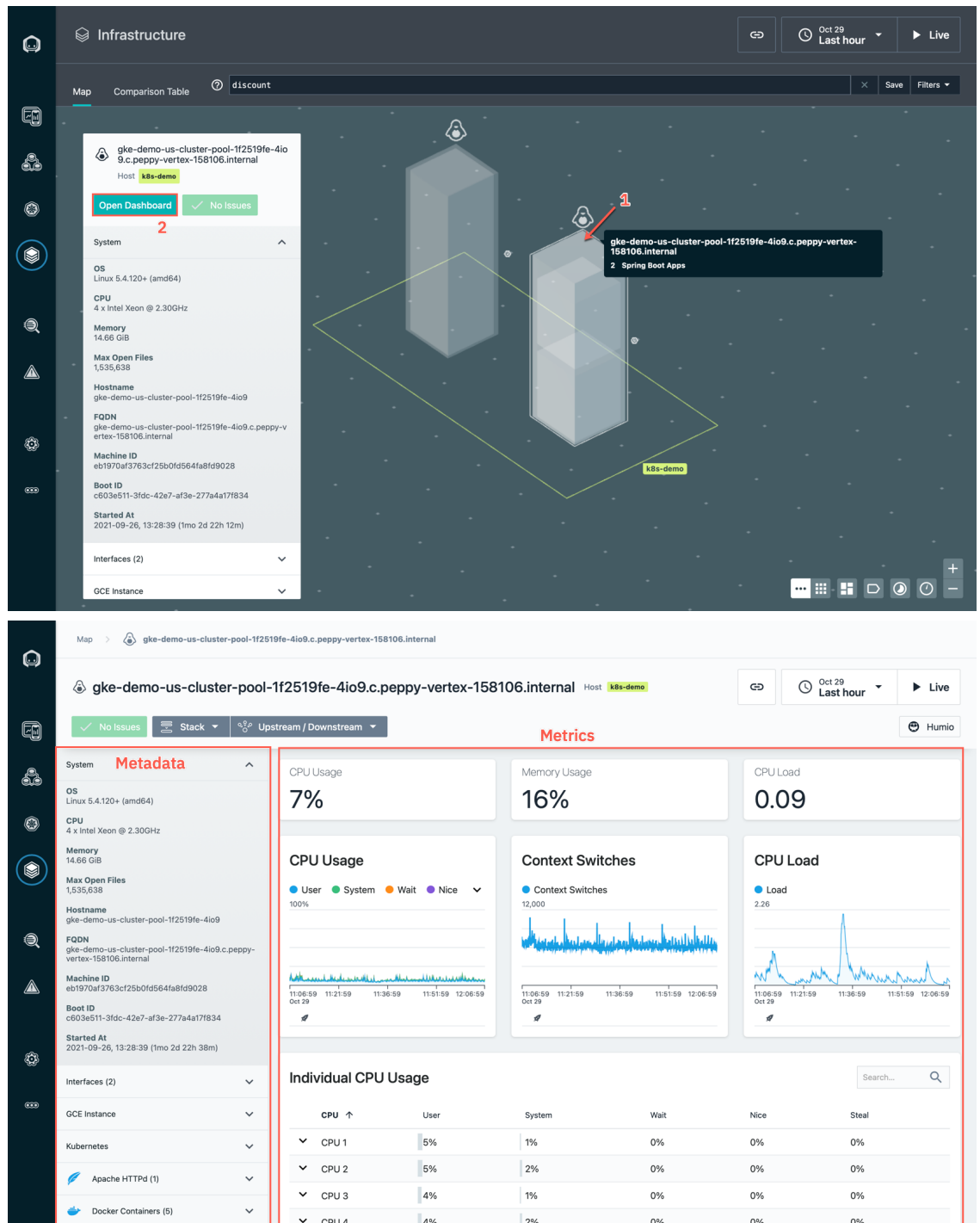
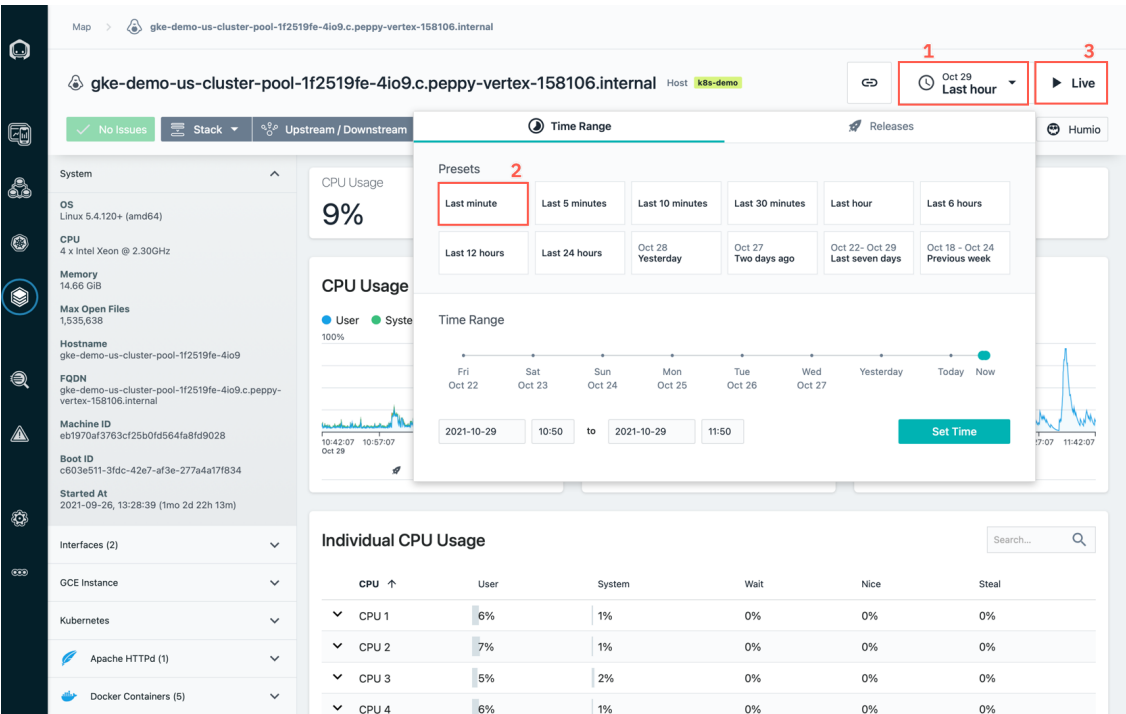## 1.3 Adjusting the time range**

Actions	Click the time range dropdown (1). Then, click **Last minute** button (2) and **Live** button (3). After a while, when graphs refresh with 1-second granularity, set time range back to **Last hour**
Narration	Instana streams 1 second resolution metrics in real time.

Screenshot

2. Context

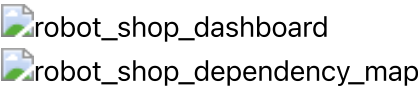| 2.1	**Automated discovery and dependency mapping**
Actions	Click the **Dependencies** tab.
Narration	Context is the interrelated conditions in which something exists or occurs; context puts things into perspective. Understanding the context of the observability data is essential with modern distributed applications where there are extensive dependencies between components. What you can see here 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. This provides the necessary context to understand how different components affect each other. We can see how requests are moving through the application in real-time. Instana can 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 dependency map when something like this happens, so let me walk you through what it looks like from the Site Reliability Engineer (SRE) or IT Operator's point of view when an incident occurs.
Screenshot	robot_shop_dashboard robot_shop_dependency_map

3. Working with events and incidents

3.1	**Automatically assessing events and alerts**

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

Narration	We've got 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 Pager Duty, 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 on incidents if a problem is likely to affect end users. Let's go into the details of this incident.

Screenshot	sidebar_menu

3.2	**Inspecting auto-correlated incident details**

Action	Click 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 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.

Screenshots	event_page Incident details screen: incident_details_screen

3.3	**Understand the incident**

Actions	Under **Related Events**, click on the event that says **Sudden increase in the number of erroneous calls**. In the end, click **Analyze Calls** button.

Narration	It looks like the abnormal termination of the MySQL database caused the problem. It shows how one data store issue rippled out to affect several directly and indirectly connected services. Instana's automatic root cause analysis uses the relationship information from the *Dynamic Graph* to accurately collate the individual issues into one incident. This helps eliminate alert storms, providing your DevOps engineers and SREs with a single notification of actionable information to enable them to promptly restore normal service. Let's look at some related traces for this.

3.3	Understand the incident

Screenshot	events

4. Unbounded Analytics

4.1	Examine call details

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

Narration	Now, we moved to the Analytics view. You can see, how Instana UI allows for easy navigation between different views, keeping the time span and context. At the top, you can see the Filter that was applied to all collected traces. All filtered requests are grouped by endpoint [*In this case it is database CONNECT exposed my MySQL server*]. There is only one endpoint here, but if there were multiple, you'd see a list. Endpoints are automatically discovered and mapped by Instana. We can go into the details for each erroneous call to MySQL via this endpoint (CONNECT).

Screenshot	endpoint_connect

4.2	View call via a visual dashboard

Action	--

Narration	Clicking on an individual call takes us to a view of the call in the 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.

Screenshot	call_timeline

4.3	Understand the impact and source of the incident

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

4.3	Understand the impact and source of the incident
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.

Screenshot	connect

Go to top

5. Confirming incident resolution was successful

5.1	Metrics for the Robot Shop have returned to normal
Actions	In the top-right corner set the timeframe so it begins at 30 minutes past the hour and ends at 45 minutes past the hour. Click **Set time** 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.

Screenshot	robot_shop_summary_end

Go to top

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, as well as latency, has dropped.

The problem with the Robot Shop has been fixed and the application's normal state was restored!

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.

Go to top

Call to Action Guidelines

Successful demo should be closed with the **Call to Action** - getting the client to agree on the next steps that would progress the opportunity.

1. Create a slide or whiteboard to help the client outline the next steps. You could also use Mural or another whiteboarding platform. The goal is to let the client drive the conversation.

2. Proposed next steps may include:

   - a deep dive analysis of the client's current APM to identify gaps and understand what the client is attempting to achieve. (Client drives this part along with the seller.)
   - Instana SaaS trial or Proof-of-Concept if warranted.

3. The output of the analysis is a solution brief that is co-authored with the client. The solution brief outlines the problem and Instana's projected solution.

4. Instana's technical team follows up after the solution brief is completed. They will have a better view of the projected solution. This drives a mutual closing plan with a pricing agreement.

5. The ultimate goal is product launch at the client.

Go to top