Oracle® Banking Microservices Architecture

Observability User Guide

Release 14.7.3.0.0
F94420-01
February 2024

Oracle Banking Microservices Architecture Observability User Guide, Release 14.7.3.0.0

F94420-01

Copyright © 2018, 2024, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Contents

Preface

Purpose v
Audience v
Documentation Accessibility v
Diversity and Inclusion vi
Related Resources vi
Conventions vi
Acronyms and Abbreviations vi
Prerequisites vi
General Prevention vii
Best Practices vii

1 Observability Improvements using Zipkin Traces

1.1 Setting Zipkin Server 1-1
1.2 Troubleshooting Zipkin 1-1
1.3 Zipkin Issues 1-5
1.3.1 Application Service is not Registered 1-5
1.3.2 404 Error 1-7
1.3.3 Unable to Change Zipkin Default Port Number 1-7

2 Observability Improvements Logs using ELK Stack

2.1 Introduction 2-1
2.2 Architecture 2-2
2.3 Setting up ELK Stack 2-2
2.3.1 Run ELK Stack 2-3
2.3.1.1 Start Elastic Search 2-4
2.3.1.2 Setup and Start Logstash 2-4
2.3.1.3 Setup and Start Kibana 2-5
2.3.2 Access Kibana 2-6
2.3.3 Kibana Logs 2-7

3 Health Checks

3.1 Discovery Health Check 3-1
3.2 Actuator Health Indicator Endpoint 3-1
3.2.1 Generic Service 3-1
3.2.2 Kafka Consumers and Producers 3-4

4 Troubleshooting Kafka Issues

4.1 Kafka Health 4-1
4.1.1 Verify Kafka Health 4-1
4.1.2 Verify Zookeeper Health 4-1
4.2 Prometheus and Grafana 4-1
4.2.1 Prometheus Setup 4-2
4.2.2 JMX-Exporter Setup 4-2
4.2.3 Grafana Setup 4-2
4.2.4 Prometheus Metrics 4-3

5 Troubleshooting Flyway Issues

5.1 Failed Migrations 5-1
5.1.1 Success Column Verification 5-1
5.1.2 Migration Checksum Mismatch for a Version 5-1
5.1.3 Placeholder errors 5-1

Index

Preface

- Purpose
- Audience
- Documentation Accessibility
- Diversity and Inclusion
- Related Resources
- Conventions
- Acronyms and Abbreviations
- Prerequisites
- General Prevention
- Best Practices

Purpose

This guide helps to use the tools that enable users to observe the Oracle Banking Microservices Architecture suite of products better.

The sections provide tools that can enable a user to:

- Observe the spans associated with various API calls and the response of each API.
- Troubleshoot Kafka better.
- Aggregate logs and interpret out of log searches.

This guide also describes recommended tools to enhance monitoring and observability aspects of the Oracle Banking Microservices Architecture products.

Audience

This guide is intended for the implementation teams.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Resources

The related documents are as follows:

- *Oracle Banking Getting Started User Guide*
- *Troubleshooting Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Acronyms and Abbreviations

The list of the acronyms and abbreviations that you are likely to find in the guide are as follows:

Table 1 Acronyms and Abbreviations

Abbreviation	Description
API	Application Programming Interface
ELK	Elasticsearch Logstash Kibana
UI	User Interface
URL	Uniform Resource Locator

Prerequisites

The prerequisites are as follows:

- Basic understanding of Eventing platform.
- Basic understanding application log analysis using tools.
- Basic understanding DB changes.

General Prevention

Do not make any changes to Flyway scripts manually.

Best Practices

The best practices are as follows:

- It is ideal to have ELK stack installed on a separate VM outside the product VMs to ensure flow of logs in case of app crash.
- Log levels can be adjusted to INFO and above to enable relevant logs to flow in.
- Verify all Kafka settings as per User Troubleshooting Guide before the health check.

1

Observability Improvements using Zipkin Traces

This topic describes the troubleshooting procedures using the Zipkin Traces.

- Setting Zipkin Server
  This topic provides the information to install and setup Zipkin server.
- Troubleshooting Zipkin
  This topic provides the systematic instructions to troubleshooting using Zipkin Traces.
- Zipkin Issues
  This topic describes the various issues faced in Zipkin.

1.1 Setting Zipkin Server

This topic provides the information to install and setup Zipkin server.

Zipkin works as an independent application, and it can be downloaded as a runnable jar from the official website of Zipkin https://zipkin.io/ The latest version of Zipkin needs a Java version above 8.

The direct download link of jar is https://search.maven.org/remote_content?g=io.zipkin&a=zipkin-server&v=LATEST&c=exec

- The downloaded jar can be executed using the java –jar JAR_NAME command.
- The configuration of Zipkin can be done environment variables. The port of the Zipkin can be set using QUERY_PORT environment variable.
- The application starts on the port number assigned for QUERY_PORT environment variable or its default value of 9411. The web UI of Zipkin can be accessed at http://localhost:PORT.

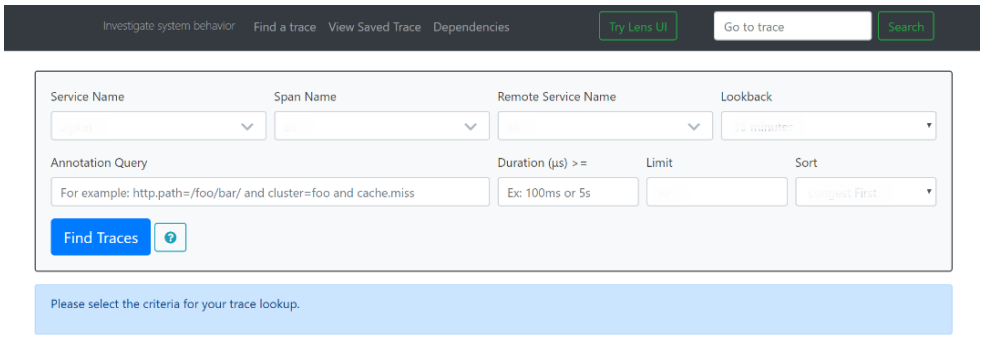
1.2 Troubleshooting Zipkin

This topic provides the systematic instructions to troubleshooting using Zipkin Traces.

1. Launch the Zipkin URL.

   The **Layout of Zipkin** screen displays.

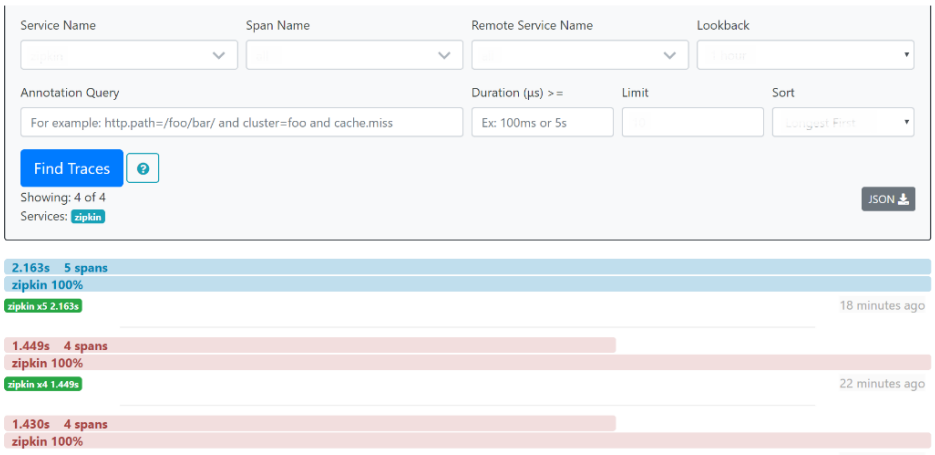
Figure 1-1 Layout of Zipkin



2. Use **Search** to find the traces of required API calls and services.

   The **List of Traces** screen displays.

Figure 1-2 List of Traces

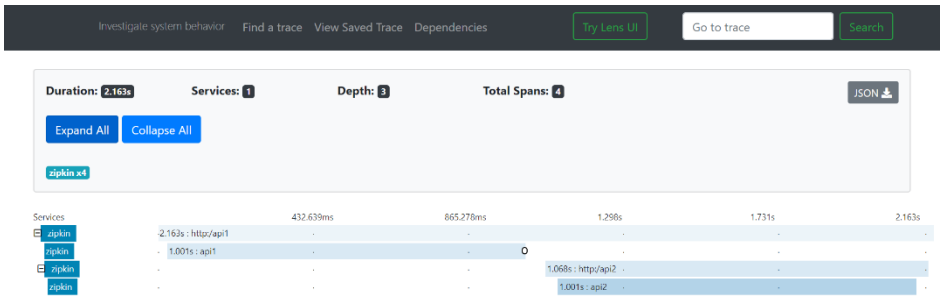


> **Note:**
>
> The search options given in the user interface are self-explanatory, and there is another UI option (**Try Lens UI**). It is given a different user interface with the same functionality. The list of the traces can be seen as shown in the above screen. Error API calls are made to showcase how to track errors. The blue listing shows the successful API hits, and the red listing indicates the errors. Each block indicates a single trace in the listing.

3. Open an individual trace to view the details of the trace.

   The **Individual Trace** screen displays.

Figure 1-3 Individual Trace

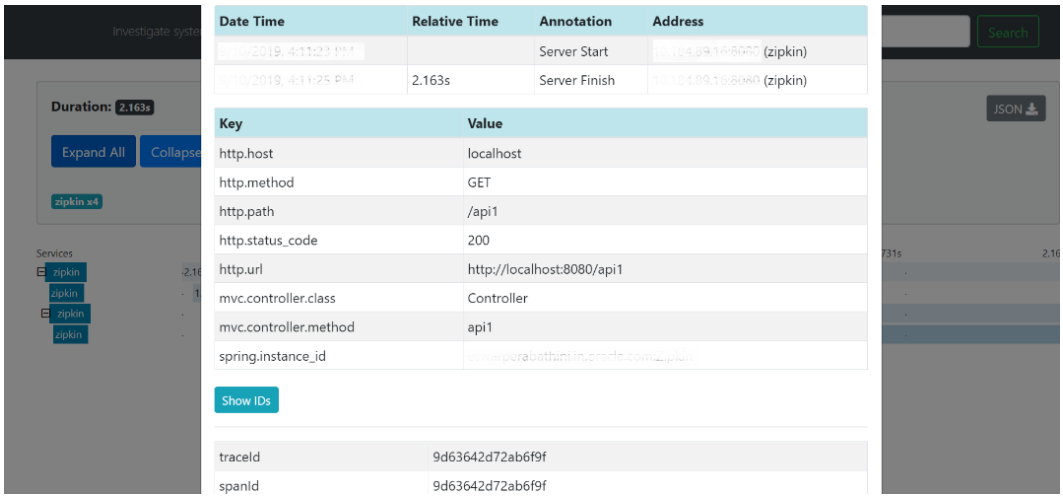


> **Note:**
>
> **Individual Trace** describes the time taken for each block. As the two custom spans are created inside two service calls, user can find a total of four blocks. The time taken for an individual block is shown in the above screen.

4. Click an individual block to display the details.

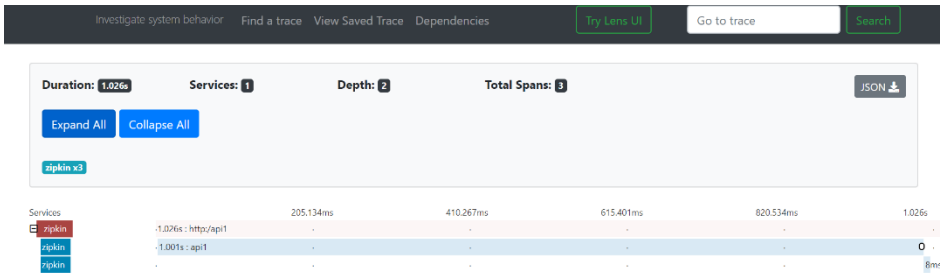
   The **Details of Individual Block** screen displays.

Figure 1-4 Details of Individual Block



> **Note:**
>
> The user can also view the span block details and logging events in the Zipkin UI as small circular blocks. An example of an error log is shown in the below screen.

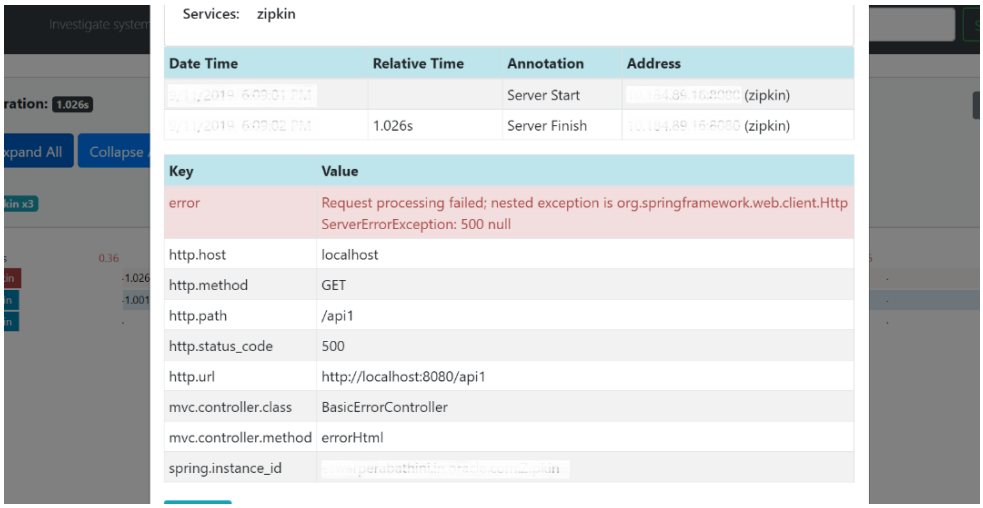
Figure 1-5 Sample Error Log



5. Click the error portion to get clear details and place of the error.

   The **Details of Error** screen displays.

Figure 1-6 Details of Error

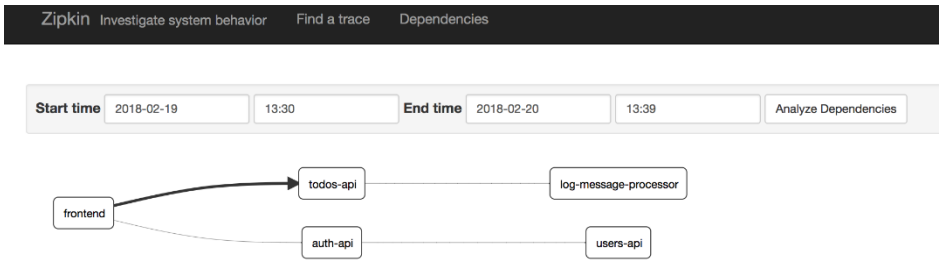


> **Note:**
>
> If the Lens UI is used in Zipkin, the above figures are not applicable but are relatable to the Lens UI as well. Traces of the application can be found using TraceId. The TraceId can be found in the debug logs of the deployment when spring-cloud-sleuth is included in the dependencies (included in spring-cloud-starter-Zipkin dependency).

6. Click **Dependencies** to get the dependency graph information between micro-services.

   The **Sample Dependency Graph** displays.

Figure 1-7 Sample Dependency Graph



1.3 Zipkin Issues

This topic describes the various issues faced in Zipkin.

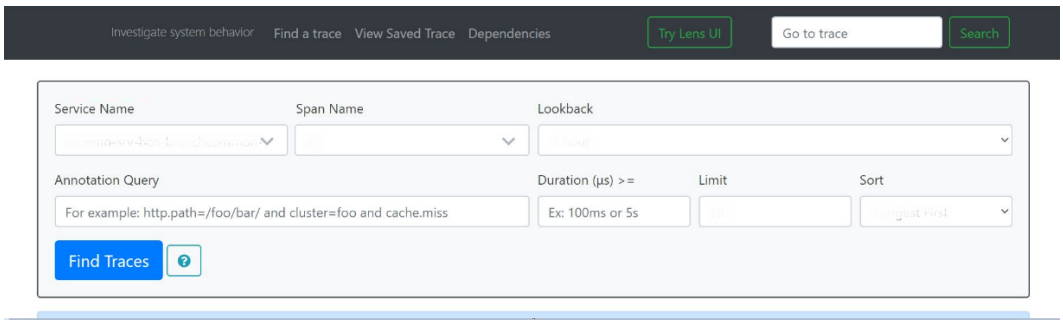
- Application Service is not Registered
  This topic describes the systematic instructions to register the application service.
- 404 Error
  This topic provides the information to troubleshoot the 404 Error in the application.
- Unable to Change Zipkin Default Port Number
  This topic provides the information to change the Zipkin Default Port Number in the application.

1.3.1 Application Service is not Registered

This topic describes the systematic instructions to register the application service.

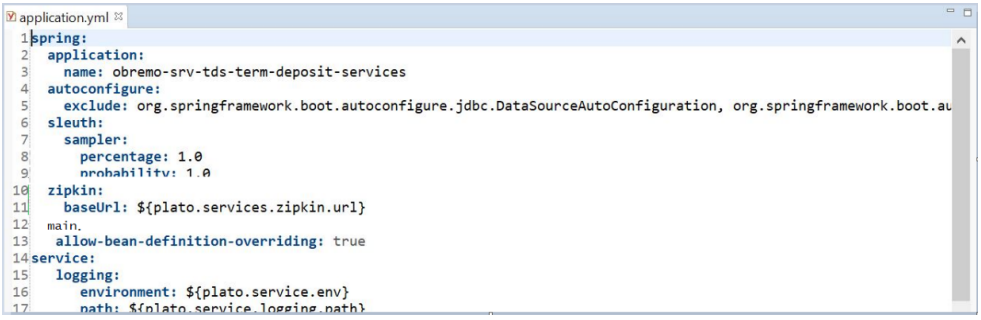
1. Check the applications that are sending the trace report to the Zipkin server from **Service Name** drop-down list.

Figure 1-8 Find Traces



2. If the required application is not listed in Zipkins, check the `application.yml` file for Zipkin base URL configuration.

Figure 1-9 Application.yml File



> **Note:**
>
> The shipped `application.yml` should have the Zipkin entry. Every service should have a `spring-cloud-sleuth-zipkin` dependency added in the build gradle file for the service to generate and send trace Id and span Id.

3. Specify the necessary values are as follows:

Compile group: org.springframework.cloud
Name: spring-cloud-sleuth-zipkin

Version: 2.1.2.RELEASE

Figure 1-10 Branch Common Services

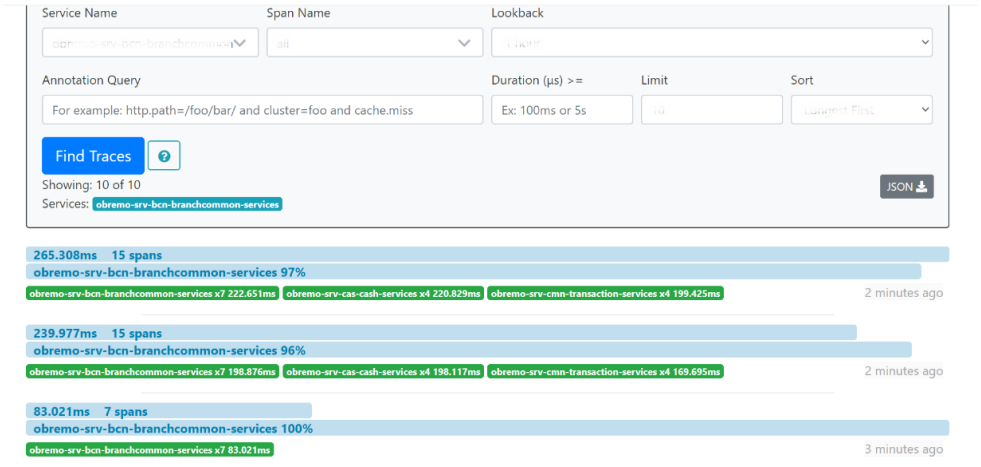
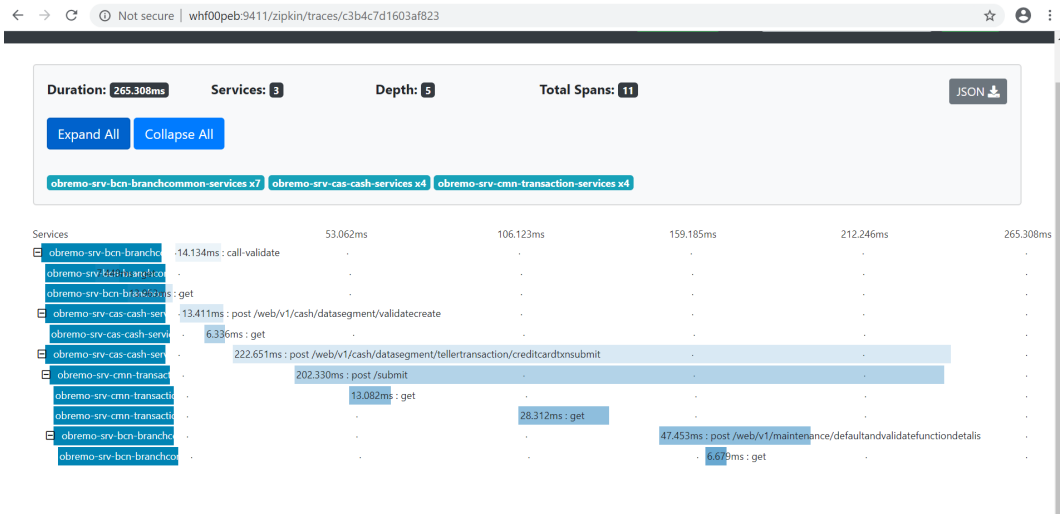


Figure 1-11 Branch Common Services Trace



1.3.2 404 Error

This topic provides the information to troubleshoot the 404 Error in the application.

If there is a 404 error, check if the **zipkin-server.jar** is running in the system where the application is deployed. To check this, execute the following command:

```
netstat -ltnup | grep ':9411'
```

The output should be like as below:

```
tcp6 0 0 :::9411 :::* LISTEN 10892/java
```

Here 10892 is the PID.

1.3.3 Unable to Change Zipkin Default Port Number

This topic provides the information to change the Zipkin Default Port Number in the application.

Zipkin default port number is not editable.

Hence, make sure that the port 9411 is available to start `Zipkin-server.jar file`.

2

Observability Improvements Logs using ELK Stack

This topic describes the troubleshooting procedures using the ELK Stack.

The ELK Stack is a collection of the following open-source products:

- **Elasticsearch**: It is an open-source, full-text search, and analysis engine based on the Apache Lucene search engine.
- **Logstash**: Logstash is a log aggregator that collects data from various input sources, executes different transitions and enhancements, and then transports the data to various supported output destinations.
- **Kibana**: Kibana is a visualization layer that works on top of Elasticsearch, providing users with the ability to analyze and visualize the data.

These components together are most commonly used for monitoring, troubleshooting, and securing IT environments. Logstash takes care of data collection and processing, Elasticsearch indexes and stores the data, and Kibana provides a user interface for querying the data and visualizing it.

- Introduction
- Architecture
  This topic describes about architecture.
- Setting up ELK Stack
  This topic describes the systematic instruction to download, run and access the ELK Stack.

2.1 Introduction

ELK Stack was a collection of the following open-source products:

- **Elasticsearch**
- **Logstash**
- **Kibana**

Elasticsearch is an open source, full-text search, and analysis engine, based on the Apache Lucene search engine. Logstash is a log aggregator that collects data from various input sources, executes different transformations and enhancements and then ships the data to various supported output destinations. Kibana is a visualization layer that works on top of Elasticsearch, providing users with the ability to analyze and visualize the data.

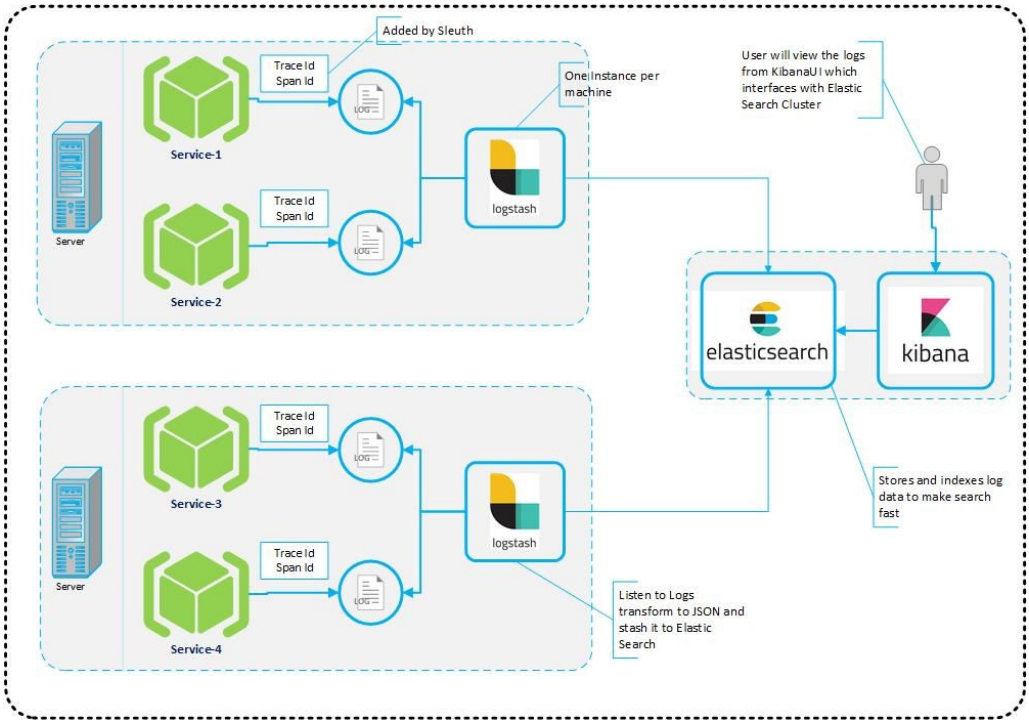
Together, these different components are most commonly used for monitoring, troubleshooting, and securing IT environments. Logstash takes care of data collection and processing, Elasticsearch indexes and stores the data, and Kibana provides a user interface for querying the data and visualizing it.

2.2 Architecture

This topic describes about architecture.

It provides a comprehensive solution for handling all the required facets.

Figure 2-1 Architecture



Spring Cloud Sleuth also provides additional functionality to trace the application calls by providing us with a way to create intermediate logging events. Therefore, Spring Cloud Sleuth dependency must be added to the applications.

2.3 Setting up ELK Stack

This topic describes the systematic instruction to download, run and access the ELK Stack.

Download ELK Stack

1. Download the Elastic search from https://www.elastic.co/downloads/elasticsearch.
2. Download the Kibana from https://www.elastic.co/downloads/kibana.
3. Download the Logstash from https://www.elastic.co/downloads/logstash.

Figure 2-2 ELK Setup

```
# Kibana is served by a back end server. This setting specifies the port to use.
#server.port: 5601

# Specifies the address to which the Kibana server will bind. IP addresses and host names are both valid values.
# The default is 'localhost', which usually means remote machines will not be able to connect.
# To allow connections from remote users, set this parameter to a non-loopback address.
server.host: "whf00peb"

# Enables you to specify a path to mount Kibana at if you are running behind a proxy.
# Use the `server.rewriteBasePath` setting to tell Kibana if it should remove the basePath
# from requests it receives, and to prevent a deprecation warning at startup.
# This setting cannot end in a slash.
#server.basePath: ""

# Specifies whether Kibana should rewrite requests that are prefixed with
# `server.basePath` or require that they are rewritten by your reverse proxy.
# This setting was effectively always `false` before Kibana 6.3 and will
# default to `true` starting in Kibana 7.0.
#server.rewriteBasePath: false

# The maximum payload size in bytes for incoming server requests.
#server.maxPayloadBytes: 1048576

# The Kibana server's name.  This is used for display purposes.
#server.name: "your-hostname"

# The URL of the Elasticsearch instance to use for all your queries.
elasticsearch.url: "http://localhost:9200"

# When this setting's value is true Kibana uses the hostname specified in the server.host
```

> **Note:**
>
> Default port for Elastic search is 9200 and the default port for Kibana is 5601.

- Run ELK Stack
  This topic describes the systematic instruction to run the ELK Stack.
- Access Kibana
  This topic describes the information to access the kibana.
- Kibana Logs
  This topic describes the information to setup, search and export the logs in Kibana.

2.3.1 Run ELK Stack

This topic describes the systematic instruction to run the ELK Stack.

Perform the following steps:

1. Run the elasticsearch.sh file present in the folder path `/scratch/software/ELK/elasticsearch-6.5.1/bin`.
2. Configure the Kibana to point the running instance of elastic search in the kibana.yml file.
3. Follow the below steps to configure the Logstash.
   a. **Input:** This configuration is used to provide the log file location for the Logstash to read from.
   b. **Filter:** This configuration is used to control or format the read operation (Line by line or Bulk read).
   c. **Output:** This configuration is used to provide the running elastic search instance to send the data for persisting.

Figure 2-3 Logstash Configuration

```
input {
  file {
    type => "java"
    path => "/scratch/Software/Weblogic_Installation/user_projects/domains//base_domain/logs/obremo-srv-cmn-transaction-services.log"
    codec => multiline {
      pattern => "Transation Ended!"
      negate => "true"
      what => "next"
    }
  }
}

filter {
  #If log line contains tab character followed by 'at' then we will tag that entry as stacktrace
  if [message] =~ "\tat" {
    grok {
      match => ["message", "^(\tat)"]
      add_tag => ["stacktrace"]
    }
  }

}

output {

  stdout {
    codec => rubydebug
  }

  # Sending properly parsed log events to elasticsearch
  elasticsearch {
    hosts => ["localhost:9200"]
  }
}
```

- Start Elastic Search
  This topic provides systematic instructions to start Elastic Search.
- Setup and Start Logstash
  This topic provides the systematic instructions to setup and start Logstash.
- Setup and Start Kibana
  This topic provides the systematic instructions to setup and start Kibana.

2.3.1.1 Start Elastic Search

This topic provides systematic instructions to start Elastic Search.

1. Navigate to Elasticsearch root folder.
2. Use nohup to start the Elasticsearch process.

```
> nohup ./bin/elasticsearch
```

2.3.1.2 Setup and Start Logstash

This topic provides the systematic instructions to setup and start Logstash.

1. Create a new **logstash.conf** file that provides the required file parsing and integration for Elasticsearch.

   logstatsh.conf:

```
#Point to the application logs
input {
 file {
  type => "java"
  path => "/scratch/app/work_area/app_logs/*.log"
  codec => multiline {
   pattern => "^%{YEAR}-%{MONTHNUM}-%{MONTHDAY} %{TIME}.*"
   negate => "true"
   what => "previous"
  }
```

```
 }
}
#Provide the parsing logic to transform logs into JSON
filter {
 #If log line contains tab character followed by 'at' then we will tag
that entry as stacktrace
 if [message] =~ "\tat" {
  grok {
   match => ["message", "^(\tat)"]
   add_tag => ["stacktrace"]
  }
 }

 #Grokking Spring Boot's default log format
 grok {
  match => [ "message",
             "(?<timestamp>%{YEAR}-%{MONTHNUM}-%{MONTHDAY}
%{TIME}) %{LOGLEVEL:level} %{NUMBER:pid} --- \[(?<thread>[A-Za-z0-9-]+)\]
[A-Za-z0-9.]*\.(?<class>[A-Za-z0-9#_]+)\s*:\s+(?<logmessage>.*)",
             "message",
             "(?<timestamp>%{YEAR}-%{MONTHNUM}-%{MONTHDAY} %{TIME}) %
{LOGLEVEL:level} %{NUMBER:pid} --- .+? :\s+(?<logmessage>.*)"
           ]
 }
  # pattern matching logback pattern
  grok {
         match =>
 { "message" => "%{TIMESTAMP_ISO8601:timestamp}\s+%{LOGLEVEL:severity}\s+\
[%{DATA:service},%{DATA:trace},%{DATA:span},%{DATA:exportable}\]\s+\[%
{DATA:environment}\]\s+\[%{DATA:tenant}\]\s+\[%{DATA:user}\]\s+\[%
{DATA:branch}\]\s+%{DATA:pid}\s+---\s+\[%{DATA:thread}\]\s+%{DATA:class}
\s+:\s+%{GREEDYDATA:rest}"
 }
   }
   #Parsing out timestamps which are in timestamp field thanks to
previous grok section
   date {
    match => [ "timestamp" , "yyyy-MM-dd HH:mm:ss.SSS" ]
   }
  }
  #Ingest logs to Elasticsearch
  output {
   elasticsearch { hosts => ["localhost:9200"] }
   stdout { codec => rubydebug }
  }
```

2. Start the Logstash process using below command.

```
>nohup ./bin/logstash -f logstash.conf
```

2.3.1.3 Setup and Start Kibana

This topic provides the systematic instructions to setup and start Kibana.

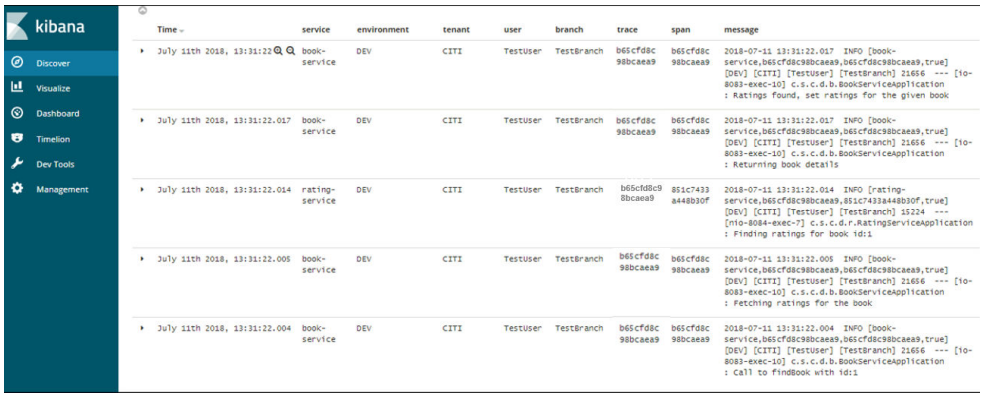
1. Navigate to the **kibana.yml** available under `<kibana_setup_folder>/config`.
2. Modify the file to include the below:

```
#Uncomment the below line and update the IP address to your host
machine IP.
server.host: "xx.xxx.xxx.xx"
#Provide the elasticsearch url. If this is running on the same
machine then you can use the below config as is
elasticsearch.url: "http://localhost:9200"
```

3. Start the Kibana process using the below command.

```
>nohup ./bin/kibana
```

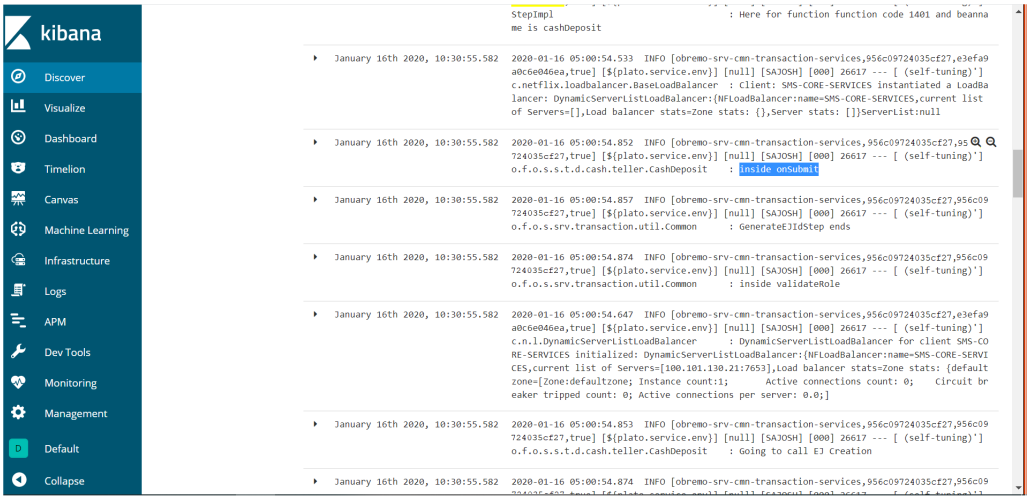
Figure 2-4 Kibana Dashboard



2.3.2 Access Kibana

This topic describes the information to access the kibana.

Figure 2-5 Access Kibana



2.3.3 Kibana Logs

This topic describes the information to setup, search and export the logs in Kibana.

Setup Dynamic Log Levels in Oracle Banking Microservices Architecture Services without Restart

The plato-logging-service is dependent on two tables, which needs to be present in the PLATO schema (JNDI name: jdbc/PLATO). The two tables are as follows:

- **PLATO_DEBUG_USERS**: This table contains the information about whether the dynamic logging is enabled to a user for a service. The table will have records, where DEBUG_ENABLED values for a user and a service have values **Y** or **N**, and depending on that plato-logger will enable dynamic logging.

Figure 2-6 PLATO_DEBUG_USERS

	ID	DEBUG_ENABLED	SERVICE_CODE	USER_ID
1	2	Y	plato-logger-ref	
2	3	Y	platoref	

- **PLATO_LOGGER_PARAM_CONFIG**: This table contains the key-value entries of different parameters that can be changed at runtime for the dynamic logging.

Figure 2-7 PLATO_LOGGER_PARAM_CONFIG

	ID	MODIFY_FIELD	PARAM_NAME	PARAM_VAL
1	3	N	LOG_PATH	C:\Oracle\Middleware\Oracle_Home\user_projects\domains\base_domain\logs
2	2	N	LOG_LEVEL	INFO
3	1	N	LOG_MSG_WITH_TIME	Y

The values that can be passed are as follows:

  - **LOG_PATH**: This specifies a dynamic logging path for the logging files to be stored. Changing this in runtime, changes the location of the log files at runtime. If this value is not passed then by default, the LOG_PATH value is taken from the -D parameter of `plato.service.logging.path`.
  - **LOG_LEVEL**: The level of the logging can be specified on runtime as **INFO** or **ERROR** etc. The default value of this can be set in the `logback.xml`.
  - **LOG_MSG_WITH_TIME**: Making this **Y** appends the current date into the log file name. Setting the value of this as **N** cannot append the current date into the filename.

Search for Logs in Kibana

Search logs in Kibana using https://www.elastic.co/guide/en/kibana/current/search.html.

Export Logs for Tickets

Perform the following steps to export logs:

1. Click **Share** from the top menu bar.

2. Select the **CSV Reports** option.
3. Click **Generate CSV** button.

3
Health Checks

This section describes the possible approaches to monitor health of Oracle Banking Microservices Architecture services.

- Discovery Health Check
  This topic describes about the health status of all the registered services and their instances.
- Actuator Health Indicator Endpoint
  This topic describes about the Health Status of the Endpoint

3.1 Discovery Health Check

This topic describes about the health status of all the registered services and their instances.

Figure 3-1 Discovery Health Check

CONFIG-SERVICE	n/a (1)	(1)	UP (1) - whf00fpr.in.oracle.com:config-service:7002
FEED-DOMAIN1-SERVICES	n/a (4)	(4)	UP (4) - localhost:feed-domain1-services:7003 , whf00fpr.in.oracle.com:feed-domain1-services:7003 , host.docker.internal:feed-domain1-services:7003 , TGEETEY-T490.in.oracle.com:feed-domain1-services:7203
PLATO-API-GATEWAY	n/a (2)	(2)	UP (2) - SAHT-T490.in.oracle.com:plato-api-gateway:7002 , whf00fpr.in.oracle.com:plato-api-gateway:7002
PLATO-BATCH-SERVER	n/a (1)	(1)	UP (1) - whf00fpr.in.oracle.com:plato-batch-server:8088
PLATO-DISCOVERY-SERVICE	n/a (1)	(1)	UP (1) - whf00fpr.in.oracle.com:plato-discovery-service:7002
PLATO-FEED-SERVICES	n/a (4)	(4)	UP (4) - host.docker.internal:plato-feed-services:7003 , whf00fpr.in.oracle.com:plato-feed-services:7003 , TGEETEY-T490.in.oracle.com:plato-feed-services:7203 , localhost:plato-feed-services:7003
PLATO-O	n/a (1)	(1)	UP (1) - whf00fpr
PLATO-O4	n/a (1)	(1)	UP (1) - DEEPMATH-T490
PLATO-RULE-SERVICE	n/a (1)	(1)	UP (1) - whf00fpr.in.oracle.com:plato-rule-service:7003
PLATO-UI-CONFIG-SERVICES	n/a (1)	(1)	UP (1) - whf00fpr.in.oracle.com:plato-ui-config-services:7002

3.2 Actuator Health Indicator Endpoint

This topic describes about the Health Status of the Endpoint

- Generic Service
- Kafka Consumers and Producers

3.2.1 Generic Service

To check the health status of any service hit the below endpoint:

```
http://<Host>:<Port>/context_path/actuator/health
```

Example:`http://localhost:8089/refapp/actuator/health`

With headers similar to:

userId: **XYZ**

appId: **PLATOREFAPP**

entityId: **DEFAULTENTITY**

branchCode: **000**

Sample Response:

```
{
    "status": "UP"
}
```

To get more detailed health status add following property:

`management.endpoint.health.show-details=always`

Sample Response:

```
{
    "status": "UP",
    "components": {
        "binders": {
            "status": "UP",
            "components": {
                "kafka": {
                    "status": "UP"
                }
            }
        },
        "clientConfigServer": {
            "status": "UP",
            "details": {
                "propertySources": [
                    "refapp-jdbc"
                ]
            }
        },
        "db": {
            "status": "UP",
            "components": {
                "PLATO_LOGGER_DS": {
                    "status": "UP",
                    "details": {
                        "database": "Oracle",
                        "validationQuery": "isValid()"
                    }
                },
                "dataSource": {
                    "status": "UP",
                    "details": {
                        "database": "Oracle",
```

```
                        "validationQuery": "isValid()"
                    }
                }
            }
        },
        "discoveryComposite": {
            "status": "UP",
            "components": {
                "discoveryClient": {
                    "status": "UP",
                    "details": {
                        "services": [
                            "plato-feed-services",
                            "plato-api-gateway",
                            "plato-rule-service",
                            "refapp"
                        ]
                    }
                },
                "eureka": {
                    "description": "Remote status from Eureka server",
                    "status": "UP",
                    "details": {
                        "applications": {
                            "PLATO-API-GATEWAY": 1,
                            "PLATO-RULE-SERVICE": 1,
                            "REFAPP": 1,
                            "PLATO-FEED-SERVICES": 4,
                        }
                    }
                }
            }
        },
        "diskSpace": {
            "status": "UP",
            "details": {
                "total": 248031522816,
                "free": 81710915584,
                "threshold": 10485760,
                "exists": true
            }
        },
        "hystrix": {
            "status": "UP"
        },
        "ping": {
            "status": "UP"
        },
        "refreshScope": {
            "status": "UP"
        }
    }
}
```

3.2.2 Kafka Consumers and Producers

To check the health status of kafka consumers and producers hit the following endpoint: `http://<Host>:<Port>/context_path/actuator/health`

To stop discovery service from routing requests to kafka consumers or producers when connection to kafka is not successful, following flag needs to be set: **`eureka.client.healthcheck.enabled=true`**

4
Troubleshooting Kafka Issues

This topic describes the troubleshooting procedures for the Kafka issues.

- Kafka Health
  This topic describes the troubleshooting procedures for the Kafka Health.
- Prometheus and Grafana
  This topic describes about the Troubleshooting Kafka issues using Prometheus and Grafana.

4.1 Kafka Health

This topic describes the troubleshooting procedures for the Kafka Health.

- Verify Kafka Health
- Verify Zookeeper Health

4.1.1 Verify Kafka Health

Run the below command and verify: **`$ netstat -tlnp | grep :9092`**

> **Note:**
>
> 9092 is default port of kafka

4.1.2 Verify Zookeeper Health

Kafka instance will not start if Zookeeper is not yet started.

1. Run the below command and verify.
   ```
   $ netstat -tlnp | grep :2181 (2181 is default port of zookeeper)

   tcp6 0 0 :::2181 :::* LISTEN 19936/java
   ```
2. To debug, check if the permissions of Kafka log folder are correct.
   The log folder path can be found by looking at the value of the property `log.dirs` in the `server.properties` file of Kafka installation.

4.2 Prometheus and Grafana

This topic describes about the Troubleshooting Kafka issues using Prometheus and Grafana.

- Prometheus Setup
- JMX-Exporter Setup
- Grafana Setup

- Prometheus Metrics

4.2.1 Prometheus Setup

Prometheus is an open-source project, which helps monitoring of the applications metrics. It is widely used for the monitoring of Kafka and its metrics. The installer for Prometheus can be downloaded Prometheus from https://prometheus.io/download/.

4.2.2 JMX-Exporter Setup

A JMX-Exporter application is used to integrate with the Kafka broker as a Java agent to expose the values of JMX MBeans as an API. The JMX-Exporter is used by the Prometheus to fetch the values of the JMX metrics.

Perform the following steps:

1. Download the latest jmx_prometheus_javaagent jar file from the maven repository in the Kafka directory along with the bin, config directories.

> **Note:**
>
> This can be used to monitor consumer_lagthttps://repo1.maven.org/maven2/io/prometheus/jmx/jmx_prometheus_javaagent/0.15.0/jmx_prometheus_javaagent-0.15.0.jar

2. Set the *KAFKA_OPTS* variable to the desired value to execute the jar as a java agent `export KAFKA_OPTS="$KAFKA_OPTS -javaagent:$PWD/jmx_prometheus_javaagent-0.15.0.jar=7071:$PWD/kafka-0-8-2.yml"`

> **Note:**
>
> You can choose the port according to our preference.

3. Restart Kafka Broker.

4.2.3 Grafana Setup

Perform the following steps:

1. Download Grafana from https://grafana.com/grafana/download in the stand-alone application mode, and extract its contents.
2. Go to the bin folder in the extracted contents, and start the Grafana server.

> **Note:**
>
> Grafana should start on the default port 3000 (HOST: 3000). The default user and password for Grafana are admin/admin.

Perform the following steps to integrate Grafana with the Prometheus instance installed:

3. Click on the Grafana logo to open the sidebar.
4. Click **Data Sources** in the sidebar.
5. Choose **Add New**.
6. Select **Prometheus** as the data source.
7. Click **Add** to test the connection and to save the new data source.

4.2.4 Prometheus Metrics

The Prometheus Metrics are as follows:

- process_cpu_seconds_total.
- http_request_duration_seconds.
- node_memory_usage_bytes.
- http_requests_total.
- process_cpu_seconds_total.

5
Troubleshooting Flyway Issues

This topic describes the troubleshooting procedures for the flyway issues.

- Failed Migrations
  This topic describes about the Troubleshooting procedures for failed migrations.

5.1 Failed Migrations

This topic describes about the Troubleshooting procedures for failed migrations.

- Success Column Verification
- Migration Checksum Mismatch for a Version
- Placeholder errors

5.1.1 Success Column Verification

Perform the following steps for the success column verification:

1. Check the f`lyway_schema_history` table to identify the migration record with success column as ‘0’.
2. Delete the record with status as ‘0.
3. Restart deployment.

5.1.2 Migration Checksum Mismatch for a Version

Perform the following steps for the success column verification:

1. Make sure that the flyway script is not manually updated before deployment.
2. If yes, then replace with original and restart deployment.

5.1.3 Placeholder errors

Pass the placeholder values using `setUserOverrides.sh` in Weblogic. Alternatively, these issues can be debugged from Weblogic console during deployment. In addition, the application specific logs can be verified for further inputs.

Index

Numerics

404 Error, *1-7*

A

Access Kibana, *2-6*
Application Service is not Registered, *1-5*
Architecture, *2-2*

D

Discovery Health Check, *3-1*

F

Failed Migrations, *5-1*

G

Generic Service, *3-1*
Grafana Setup, *4-2*

H

Health Checks, *3-1*

I

Introduction, *2-1*

J

JMX-Exporter Setup, *4-2*

K

Kafka Consumers and Producers, *3-4*
Kafka Health, *4-1*
Kibana Logs, *2-7*

M

Migration Checksum Mismatch for a Version, *5-1*

O

Observability Improvements Logs using ELK Stack, *2-1*
Observability Improvements using Zipkin Traces, *1-1*

P

Placeholder errors, *5-1*
Prometheus and Grafana, *4-1*
Prometheus Metrics, *4-3*
Prometheus Setup, *4-2*

R

Run ELK Stack, *2-3*

S

Setting up ELK Stack, *2-2*
Setting Zipkin Server, *1-1*
Setup and Start Kibana, *2-5*
Setup and Start Logstash, *2-4*
Start Elastic Search, *2-4*
Success Column Verification, *5-1*

T

Troubleshooting Flyway Issues, *5-1*
Troubleshooting Kafka Issues, *4-1*
Troubleshooting Zipkin, *1-1*

U

Unable to Change Zipkin Default Port Number, *1-7*

V

Verify Kafka Health, *4-1*
Verify Zookeeper Health, *4-1*