

CA Jarvis - 2.3

CA Jarvis

Date: 08-Jul-2019

This Documentation, which includes embedded help systems and electronically distributed materials, (hereinafter referred to as the "Documentation") is for your informational purposes only and is subject to change or withdrawal by CA at any time. This Documentation is proprietary information of CA and may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA.

If you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all CA copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is CA.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Copyright © 2019 Broadcom. All Rights Reserved. The term "Broadcom" refers to Broadcom Inc. and/or its subsidiaries. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Table of Contents

Release Notes	11
New Features	11
Unified API for CA Jarvis and CA Jarvis Dashboard	11
Export Data from Jarvis	11
Health Check API	11
API to Delete Tenants	11
Release Comparison	11
Known Issues	12
Product Accessibility Features	12
Third-Party Software Agreements	13
Release Notes 2.3.1	17
New Features	17
Health Check API	17
API to Delete Tenants	17
Release Comparison	17
Known Issues	18
Fixed Issues	18
Product Accessibility Features	18
Third-Party Software Agreements	18
 Getting Started	 23
What Is CA Jarvis?	23
Why Jarvis? A Sample Scenario:	23
Architecture	23
Components of CA Jarvis	24
Jarvis Elasticsearch Utilities	25
Data Purge	26
Index Rollover (size based)	26
Force Merge	27
Snapshot and Restore	27
 Installing	 28
Docker-Based Installation	28
Docker Swarm Installation	28
Prerequisites	28
Hardware Requirements	28
Installation Instructions	29

Docker Swam Installation Types	29
Vertical Scaling	32
Docker Volume	32
Clear Docker Swarm	32
Install CA Jarvis Dashboard for CouchDB	33
Prerequisites	33
InstallAnywhere-Based Installation	33
CA Jarvis InstallAnywhere - Single Node	34
Prerequisite	34
Install CA Jarvis - Non-SSL	34
Install CA Jarvis - One Way SSL	42
CA Jarvis InstallAnywhere - Multi Node	51
Prerequisite	51
CA Jarvis IA - One Way SSL	51
CA Jarvis - Non SSL	63
Installation of CA Jarvis Dashboard	73
Installation of CA Jarvis Dashboard on a Single Node (Without High-Availability Failover)	73
Installation of CA Jarvis Dashboard on a Single Node (With High-Availability Failover)	77
Installation of CA Jarvis Dashboard on a Single Node: Couchdb (Without High-Availability Failover)	82
Upgrading and Migrating	87
Upgrading	87
Upgrade CA Jarvis from 2.2 to 2.3	87
(Single Node) Upgrade CA Jarvis from 2.2 to 2.3	87
(Multi-Node) Upgrade CA Jarvis from 2.2 to 2.3	91
Migrating and Exporting	96
Export CA Jarvis Data	97
Prerequisites	97
Export Data from CA Jarvis	97
Configuring	99
Secure Sockets Layer (SSL)	99
Set Up SSL Certificates	99
Set Up SSL Certificates Obtained From Certificate Authority (Recommended in Production Environment)	99
Set Up Self-Signed Certificates (Recommended only for Testing Purpose in Non-Production Environment)	101
Reload SSL Certificates	104
Capture Usage Metrics Using Jmetric	105
Sample JMetrics DocType Schema	108
DAS Queries	110

Back Up and Restore Elasticsearch Data	111
Create Snapshot and Back Up It to the S3 Registry	111
Follow these steps:	111
Back Up Elasticsearch Data Using File System	112
Back Up Elasticsearch Data (Docker Swarm Multiple Nodes)	112
Restore Elasticsearch Data	116
Restore Elasticsearch Data on an Existing Cluster	116
Restore Elasticsearch Data on a New Cluster	117

Using 119

API Reference	119
Product Onboarding API	119
Create a Product	120
Update a Product	122
Get All Products	124
Get Specific Product	125
Tenant Onboarding API	126
Create a Tenant	126
Update a Tenant	129
Get All Tenants of a Product	131
Get Specific Tenant Details	133
Delete a Tenant	134
Mapping API	134
Create Document Type Definition	146
Get All Document Type Definitions	149
Get Specific Document Type Definition	153
Update Document Type	157
Data Ingestion API	160
Data Access Service API (DAS)	162
JSON Data Representation	164
Jarvis Query Language (JQL)	164
Filter Operations	166
Functions	167
Grouping Functions	167
Metric Functions	168
Nested Functions	169
Response Formats	169
Queryable Metadata	171
DocView API	172
Create a Doc View	172
Update a Doc View	173
Get a Doc View	174
Get all Doc Views for a Product	174
Delete a Doc View	175
Speed Job Onboarding API	175

Onboard a Speed Job	175
Update a Speed Job	177
Batch Job Onboarding API	180
Onboard a Batch Job	180
Update a Batch Job	182
Cron Expressions	184
Health Check API	186
Check the Health of All the Jarvis Services	186
Check the Health of Individual Services	187
CA Jarvis Dashboard	188
Prerequisites for Using CA Jarvis Dashboard	189
Accessing CA Jarvis Dashboard	189
Components	189
Viewlets	190
Dashboards	190
Layout Templates	190
Charts	190
User Interface	194
Dashboards	194
Viewlets	194
Set Masthead of CA Jarvis Dashboard	195
Using Viewlets	195
Create a Viewlet	196
Manage Viewlets	201
Using Dashboards	203
Create a Dashboard	204
Manage Dashboards	205
API Reference of CA Jarvis Dashboard	208
Product Onboarding API	209
Tenant Onboarding API	211
User Onboarding API	213
Default Dashboard Onboarding	216

Integration 217

Jarvis Analytics Framework (JAF)	217
Getting Started with JAF SDK	217
What is JAF SDK?	217
JAF SDK Architecture	217
Development Phase and Production Phase	218
Set Up Development Environment	220
How to Use JAF SDK?	223
Flow of Invocation of APIs for Creating Batch Job	223
Flow of Invocation of APIs for Creating Batch Job	224
JAF API Reference	224

Batch API Reference	224
Speed API Reference	228

Troubleshooting 232

Installation	233
Error while installing Jarvis.	233
What are the configurations I should make after installing Oracle Java 1.8.0_112?	233
I am getting XMLScriptReader error while installing CA Jarvis.	234
How to start and stop Jarvis services?	234
Why I am getting bad certificate error in ESUtils log when I install Jarvis by enabling two-way SSL (ESUtils on one machine and other Jarvis components are on another machine)?	236
I am using an Elasticsearch cluster setup. What should I do if jarvis_metadata and jarvis_kron indices are not created and no document (Analytics Schema) is present in jarvis_metadata? ..	236
I am getting the bad interpreter error while running prepareMachineAsRoot.sh and jarvisInstaller.sh.	236
What are the different error codes that appear during installation on plain virtual machine and how to resolve the errors?	237
How to check which version of CA Jarvis that I am using?	238
Log Files	239
Where can I find log files for CA Jarvis?	239
Onboarding and Ingestion	239
Why I am getting the following "400 Not Found" error message with the following content when trying to onboard a product?	239
Why I am getting 5 XX errors while onboarding a product?	240
Why I am getting 4 XX errors while onboarding a product?	240
Why does the Ingestion API returns a 503 response?	240
How do I validate whether Ingestion is working fine or not?	241
Why I am getting 4 XX errors during ingestion?	241
Why my data ingestion failed?	241
I want to do onboarding and ingestion in my SSL-enabled setup. What all should I take care for SSL?	241
I am using Postman client for onboarding and ingestion in an SSL-enabled setup. What all should I take care for SSL?	242
Verifier	242
How do I check the number of documents processed by Verifier?	242
How do I validate whether Verifier is working fine or not?	242
Why the data verification failed?	243
Indexer	243
How do I check the number of documents processed by indexer?	243
Why the indexer failed?	243
ElasticSearch	243
How to check if Elastic cluster nodes are properly configured or not?	244
How to check the health of a cluster?	244
What do the colors (green, yellow, and red) indicate?:	244

Why I am facing a drop in the ingestion rate in my production environment?	245
Kafka	246
How do I check whether Kafka is clustered or not?	246
Kafka messages are not being processed by Veifier, Indexer, or Ingestion services?	246
Why I am getting 5 XX errors?	246
Why I am getting 4 XX errors?	247
Data Purge	247
Where can I find the Data Purge logs?	247
How do I get to know if Data Purge has been initiated on any product/doc_type?	247
How do I validate what retention period is used if any tenant does not have retention period defined?	247
How do I ensure that Data Purge has been successfully completed?	248
Does Data Purge delete the index to which data is currently getting ingested?	248
CA Jarvis Dashboard	248
What are the different error codes that I get while using CA Jarvis Dashboard APIs?	248

Product Accessibility Features 251

CA Jarvis

Release Notes

This release notes article explains the key features and details for the current release of CA Jarvis.



Note: For installation and software requirements, see [Installing](#).

New Features

This release includes these new features:

Unified API for CA Jarvis and CA Jarvis Dashboard

The Product and Tenant onboarding APIs of CA Jarvis can now onboard Product/Tenant in CA Jarvis Dashboard as well by providing the details in the respective API request body. This reduces the two-step process of onboarding the product/tenant in two systems.

Export Data from Jarvis

A utility is provided to export the data from CA Jarvis for a combination of Product/Tenant or Product/Tenant/DocType. The data can be extracted for a given time period as well. The data is exported in JSON format. The exported data can be archived in any external storage for future usage.

Health Check API

The health of different components of Jarvis can now be checked through a RESTful API. These APIs enable integrating Jarvis health check into any monitoring dashboard.

API to Delete Tenants

CA Jarvis now provides the capability to delete existing tenants and all the related data and metadata. This feature mainly helps Jarvis Admins to delete tenants when the trial period ends.

Release Comparison

This table compares the key features in all active releases for CA Jarvis:

Key Features	2.3	2.2	2.1	2.0
--------------	-----	-----	-----	-----

Key Features	2.3	2.2	2.1	2.0
Unified API for CA Jarvis and CA Jarvis Dashboard	yes	no	no	no
Export Data from Jarvis	yes	no	no	no
Health Check API	yes	no	no	no
CA Jarvis Dashboard	yes	yes	no	no
JAF SDK to bring your own data science	yes	yes	no	no
Jmetric	yes	yes	no	no
Data Routing	yes	yes	no	no
Schema Registry	yes	yes	no	no
SSL/TLS-based security (1-way)	yes	yes	yes (non-Docker)	yes (non-Docker)
SSL/TLS-based security (2-way)	yes	yes	yes (non-Docker)	no
Encryption for Data at rest	yes	yes	yes (non-Docker)	yes (non-Docker)
Javis Health Checker	yes	yes	yes	yes
AVRO Encoder Service	yes	no (replaced with Schema Registry)	yes	yes
Docker-based installation	yes	yes	yes	yes
Docker-Swarm-based installation (for multiple nodes)	yes	yes	no	no
Flavor-based installation	yes	yes	yes	no
Priority Processing	yes	yes	yes	no
Multi-Value Field Support	yes	yes	yes	no

Known Issues

- **DE329605 : set tenant doctype retention period**

When an onboarded tenant is deleted, retention periods at various levels are not reset to 1.

Product Accessibility Features

CA Technologies are committed to ensure that all customers, regardless of ability, can successfully use its products and supporting documentation to accomplish vital business tasks.

Third-Party Software Agreements

CA Jarvis uses the following third-party software. To read each complete license, download the [zip file](#).

- Apache Commons Validator 1.6
- Apache Curator 2.12.0
- Apache Http Components 4.5.3
- Apache httpclient 4.5.2
- Apache-log4j-extras 1.2.17
- Apache spark 2.0.2
- Apache-tomcat-8.0.30
- BDR postgresSQL 1.0.2
- Bijection 0.9.2
- Bijection-avro_2.10 0.9.2
- Commons Codec 1.9
- Commons Cli 1.4
- Commons Collections 4.1
- Core4j 0.4
- Docker Community Edition 17.03.1-CE
- Elasticsearch 5.6.5
- Google-gson 2.3.1
- Google-gson 2.8.0
- Google Maps API 3.0
- Google-map-react 0.23.0
- Google-map-react 0.24.0
- Gradle-downloadtask 3.2.0
- Guava 19.0
- Guava 21.0
- Hadoop 2.7.3

- Importlib 1.0.4
- InstallAnywhere 2017
- Jackson-annotations 2.7.4
- Jackson 2.6.5
- Jackson-core 2.7.4
- Jackson-databind 2.6.5
- Jackson-databind 2.7.4
- Java exec 1
- Java Mail 1.4.7
- Javax.mail 1.4.7
- Jersey 2.22.2
- Jersey 2.25.1
- Jersey-container-servlet-core 2.22.1
- Jest 2.0.0
- Jest 2.0.2
- Joda-time 2.9.7
- Joda-time 2.9.9
- JSON 20160212
- Json-flattener 0.2.3
- Json-path 2.2.0
- Json-sanitizer 1.1
- Junit 4.12
- Kafka 0.10.1.0
- Kafka 2.11-0.9.0.0
- Kafka_2.11 0.10.1.0
- Kafka-clients 0.9.0.0
- Kafka JVM client 0.9.0.0
- Kafkapython==1.3.5
- kafka-schema-registry 3.2.0

- Kafka-schemaregistry-client 3.2.0
- Kitesdk 1.1.0
- Log4j 2.6
- Log4j-api 2.8.2
- Log4j-core 2.8.2
- Logback-classic 1.1.1
- Logback classic 1.1.1
- Material-ui 0.17.1
- Material-ui 0.18.1
- Material-ui 0.18.6
- Mockito 2.2.15
- Mockito 2.7.13
- Moment 2.18.1
- Minimal-json '0.9.4
- Mybatis 3.2.3
- OData4J 0.7
- Oracle Java Runtime Environment (JRE) 1.8.0_131
- Pandas 0.21.0
- Pip 8.1.2
- PostgreSQL 9.4.5
- PostgreSQL 9.6.3
- PostgreSQL JDBC Driver 9.4.1212
- Python avro==1.8.2
- Python numpy 1.13.3
- Python py4j 0.10.6
- Pythondateutil==2.6.1
- Python-py 1.4.34
- Quartz 2.2.1
- Quartz-scheduler 2.2.1

- React 15.4.2
- React-dom 15.4.2
- React-leaflet 1.6.4
- React-leafletheatmap-layer 1.0.2
- React-router 3.0.2
- React-router 3.0.5
- React-sortable-hoc 0.6.3
- React-tap-eventplugin 2.0.1
- Recharts 0.21.2
- Recharts 0.22.4
- Requests==2.18.4
- Rest-utils 3.2.0
- Rpm-confluent-schema-registry== 0.1.1
- S3-repository 5.5.0
- Scipy 1.0.0
- Search-guard-ssl 5.6.5-23
- Snappy for Java 1.1.2.6
- Spring-boot 1.3.5.RELEASE
- Spring-boot-starter-web 1.3.5.RELEASE
- Spring-context-support 4.2.5
- Spring-context-support 4.2.6
- Spring-Core 4.3.5
- Spring framework 4.2.5.RELEASE
- Spring framework 4.2.6.RELEASE
- Spring framework 4.3.5.RELEASE
- Spring-test 4.2.5
- Spring TX 4.2.6
- Spring framework 3.2.8
- Swagger-UI 3.4.2

- [Testng 6.8.7](#)
- [Testng 6.11](#)
- [Transport 5.5.3](#)
- [Whatwg-fetch 2.0.2](#)

Release Notes 2.3.1

This release notes article explains the key features and details for the current release of CA Jarvis.

- [New Features](#)
 - [Health Check API](#)
 - [API to Delete Tenants](#)
- [Release Comparison](#)
- [Known Issues](#)
- [Fixed Issues](#)
- [Product Accessibility Features](#)
- [Third-Party Software Agreements](#)

New Features

This release includes these new features:

Health Check API

The health of different components of Jarvis can now be checked through a RESTful API. These APIs enable integrating Jarvis health check into any monitoring dashboard.

API to Delete Tenants

CA Jarvis now provides the capability to delete existing tenants and all the related data and metadata. This feature mainly helps Jarvis Admins to delete tenants when the trial period ends.

Release Comparison

This table compares the key features in all active releases for CA Jarvis:

Key Features	2.3.1	2.2	2.1	2.0
Health Check API	yes	no	no	no
Jmetric	yes	yes	no	no
Data Routing	yes	yes	no	no
Schema Registry	yes	yes	no	no
SSL/TLS-based security (1-way)	yes	yes	yes	yes
SSL/TLS-based security (2-way)	yes	yes	yes	no

Key Features	2.3.1	2.2	2.1	2.0
Encryption for Data at rest	yes	yes	yes	yes
Javis Health Checker	yes	yes	yes	yes
AVRO Encoder Service	yes	no (replaced with Schema Registry)	yes	yes
Priority Processing	yes	yes	yes	no
Multi-Value Field Support	yes	yes	yes	no

Known Issues

Issue	Details
DE329605: set tenant doctype retention period	When an onboarded tenant is deleted, retention periods at various levels are not reset to 1.

Fixed Issues

Issue	Fix
DE331402: DAS and ingestion API documentation are not connected in swagger	DAS and ingestion API documentation is now available in Swagger.
DE328719: Schema registry connection resource leaks	Resource leaks on Schema Registry are addressed.

Product Accessibility Features

CA Technologies are committed to ensure that all customers, regardless of ability, can successfully use its products and supporting documentation to accomplish vital business tasks.

Third-Party Software Agreements

CA Jarvis uses the following third-party software. To read each complete license, download the zip file.

- Apache Commons Validator 1.6
- Apache Curator 2.12.0
- Apache Http Components 4.5.3
- Apache httpclient 4.5.2
- Apache-log4j-extras 1.2.17

- Apache spark 2.0.2
- Apache-tomcat-8.0.30
- BDR postgresSQL 1.0.2
- Bijection 0.9.2
- Bijection-avro_2.10 0.9.2
- Commons Codec 1.9
- Commons Cli 1.4
- Commons Collections 4.1
- Core4j 0.4
- Docker Community Edition 17.03.1-CE
- Elasticsearch 5.6.5
- Google-gson 2.3.1
- Google-gson 2.8.0
- Google Maps API 3.0
- Google-map-react 0.23.0
- Google-map-react 0.24.0
- Gradle-downloadtask 3.2.0
- Guava 19.0
- Guava 21.0
- Hadoop 2.7.3
- Importlib 1.0.4
- InstallAnywhere 2017
- Jackson-annotations 2.7.4
- Jackson 2.6.5
- Jackson-core 2.7.4
- Jackson-databind 2.6.5
- Jackson-databind 2.7.4
- Java exec 1
- Java Mail 1.4.7

- Javax.mail 1.4.7
- Jersey 2.22.2
- Jersey 2.25.1
- Jersey-container-servlet-core 2.22.1
- Jest 2.0.0
- Jest 2.0.2
- Joda-time 2.9.7
- Joda-time 2.9.9
- JSON 20160212
- Json-flattener 0.2.3
- Json-path 2.2.0
- Json-sanitizer 1.1
- Junit 4.12
- Kafka 0.10.1.0
- Kafka 2.11-0.9.0.0
- Kafka_2.11 0.10.1.0
- Kafka-clients 0.9.0.0
- Kafka JVM client 0.9.0.0
- Kafkapython==1.3.5
- kafka-schema-registry 3.2.0
- Kafka-schemaregistry-client 3.2.0
- Kitesdk 1.1.0
- Log4j 2.6
- Log4j-api 2.8.2
- Log4j-core 2.8.2
- Logback-classic 1.1.1
- Logback classic 1.1.1
- Material-ui 0.17.1
- Material-ui 0.18.1

- Material-ui 0.18.6
- Mockito 2.2.15
- Mockito 2.7.13
- Moment 2.18.1
- Minimal-json '0.9.4
- Mybatis 3.2.3
- OData4J 0.7
- Oracle Java Runtime Environment (JRE) 1.8.0_131
- Pandas 0.21.0
- Pip 8.1.2
- PostgreSQL 9.4.5
- PostgreSQL 9.6.3
- PostgreSQL JDBC Driver 9.4.1212
- Python avro==1.8.2
- Python numpy 1.13.3
- Python py4j 0.10.6
- Pythondateutil==2.6.1
- Python-py 1.4.34
- Quartz 2.2.1
- Quartz-scheduler 2.2.1
- React 15.4.2
- React-dom 15.4.2
- React-leaflet 1.6.4
- React-leafletheatmap-layer 1.0.2
- React-router 3.0.2
- React-router 3.0.5
- React-sortable-hoc 0.6.3
- React-tap-eventplugin 2.0.1
- Recharts 0.21.2

- Recharts 0.22.4
- Requests==2.18.4
- Rest-utils 3.2.0
- Rpm-confluent-schema-registry== 0.1.1
- S3-repository 5.5.0
- Scipy 1.0.0
- Search-guard-ssl 5.6.5-23
- Snappy for Java 1.1.2.6
- Spring-boot 1.3.5.RELEASE
- Spring-boot-starter-web 1.3.5.RELEASE
- Spring-context-support 4.2.5
- Spring-context-support 4.2.6
- Spring-Core 4.3.5
- Spring framework 4.2.5.RELEASE
- Spring framework 4.2.6.RELEASE
- Spring framework 4.3.5.RELEASE
- Spring-test 4.2.5
- Spring TX 4.2.6
- Spring framework 3.2.8
- Swagger-UI 3.4.2
- Testng 6.8.7
- Testng 6.11
- Transport 5.5.3
- Whatwg-fetch 2.0.2

Getting Started

- [What Is CA Jarvis?](#)
- [Why Jarvis? A Sample Scenario:](#)
- [Architecture](#)
- [Components of CA Jarvis](#)
 - [Jarvis Elasticsearch Utilities](#)
 - [Data Purge](#)
 - [Index Rollover \(size based\)](#)
 - [Force Merge](#)
 - [Snapshot and Restore](#)

What Is CA Jarvis?

CA Jarvis (Jarvis) is a reference implementation of the CA Analytics Platform. Jarvis is an analytics engine, which does the following functions:

- Receives data from applications (ingestion)
- Derives insights from the data. (batch/stream processing)
- Serves the data and insights back to the applications

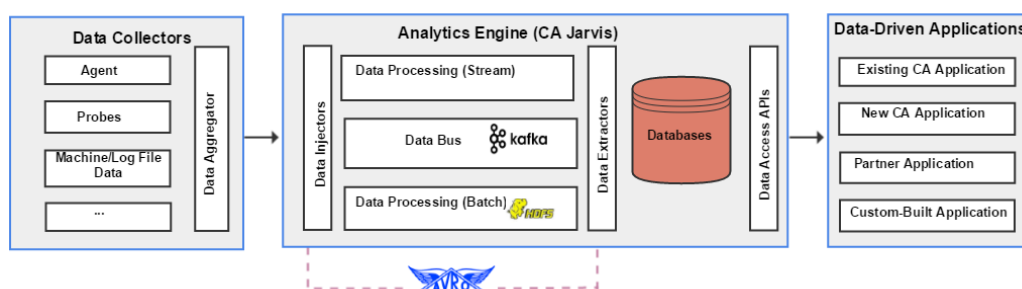
Why Jarvis? A Sample Scenario:

You are a mobile app developer and you want to identify the performance issues and failures of your app. You have the syslog and eventlog data from the remote devices. Jarvis, the analytics engine, comes into the picture now. Jarvis receives log data from your application, it then derives insights from the data, and it serves the insights back to you.

The microservices architecture makes Jarvis a suite of independently-deployable, small, modular services in which each service runs a unique process and communicates through the Kafka Message Bus. Organizations (tenants) register their analytical applications (products) with Jarvis for using it as the analytics engine. The individual microservices receive requests, process them, and generate response accordingly. These microservices communicate with each other over the internal Apache Kafka message bus of Jarvis. Each service has a primary function, for example, onboarding, ingestion, data access, indexing, and so on.

Architecture

Following is a high-level architecture diagram of Jarvis:



Components of CA Jarvis

The following section explains about the components of the CA Jarvis:

Onboarding APIs

Onboarding APIs are microservices that contain the Jarvis RESTful Onboarding API. Following are the functions of the Onboarding APIs:

- Create new product, tenant, and document type.
- Get details of a particular product, tenant, and document type.
- Update document type.

Ingestion APIs

Ingestion APIs are microservice that support bulk insertion of documents that are previously registered using the OnBoarding APIs.

Pluggable Data Science API

These APIs are responsible for registering and configuring pluggable Data Science models.

Kafka Bus

Kafka is an open source distributed message broker. Microservices can send messages to Kafka and can receive messages from it. Kafka translates a message from the formal messaging protocol of the sender (producer) to the formal messaging protocol of the receiver (consumer).

Zookeeper Registry

Zookeeper is the Topic Registry System that Jarvis uses to store the configuration files and metadata that are associated with Jarvis. For example, it stores the data (product definition, tenant definition, document mapping definitions, Avro Schema Mappings, and so on) submitted through the OnBoarding API microservice. The nodes in zookeeper are called znodes, which can store small amount of data. Microservices talk directly to znodes through the Kafka message bus to pass messages. znodes provide the ability for microservices to subscribe to topics so that when that topic gets updated, the corresponding microservice gets notified.

Hadoop HDFS

The Hadoop Distributed File System (HDFS) is Java-based file system that can store large volumes of data in a highly fault-tolerant manner. HDFS stores both raw and processed data in a partitioned manner so that the data can be accessed efficiently. Jarvis stores incoming messages to HDFS in avro format. This data storage can be used by Batch layer to produce meaningful analytical models.

Verifier and AVRO Encoder Service

Verifier reads messages from Kafka, verifies the format, and pushes the encoded methods back into Kafka.

Avro encodes the data on the Kafka Message bus as well as the data on the long term storage (Hadoop HDFS) into a compact binary format, which can be decoded by any application. Avro encoding makes the stored-data durable as the data schema is updated over time. The Avro encoder service reads the messages from the unverified Kafka topic and the corresponding meta data from the zookeeper. It converts the input messages to Avro messages and write them back to Kafka verified topic.

Doc Type Service

Doc Type service writes the Jarvis metadata into the zookeeper znodes. Doctype service intercepts the onboarding API and convert the metadata into Avro format and write it to znodes.

Spark Streaming

Spark streaming streams the data from the Kafka topic and write the raw data into HDFS.

Scheduler (Kron)

Kron scheduler is a centralized scheduling system that handles all the schedule processing for the Jarvis platform. The scheduled jobs are stored in Elasticsearch.

ES Indexer

This microservice is responsible for performing the following operations against Elasticsearch for the Jarvis Service Layer:

- Creation and updating of index definitions
- Re-indexing of indexes

Jarvis Elasticsearch Utilities

Jarvis Elasticsearch Utilities (jarvis_es_utils) contains the following services:

- Data Purge
- Index Rollover
- Force Merge
- Snapshot and Restore

Data Purge

Consider the following scenarios:

- You want to purge out raw data beyond a certain limit, so that only the data as per the licensing terms is retained.
- You want to retain data of few data types longer than other data.

In these situations, you can provide the number of days up to which the data should be kept as the retention period. For example, if Freemium customers are entitled to store data only for 45 days, the retention period (retention_period) can be set as 45. If you keep 0 as the retention_period, the Purge service will not delete any ingested data. You can set retention_period through the onboarding APIs of product and tenant and through the doc_type API.



If retention_period is set at multiple levels, the following precedence is followed while determining which value should be considered:

1. tenant-doc_type level
2. tenant level
3. doc_type level
4. product_level
5. purge default

Based on the retention period if an index contains only old data, the index gets deleted. Data Purge runs at 9 PM every day. The Data Purge logs show the status of the purge process and the retention period used. Data Purge does not delete ingestion index even if the data is older than the retention period because Data might come at any time and it gets ingested to the ingestion index. See the [API documentation](#) for more information – about how to run it at different time or time interval.

Index Rollover (size based)

The indexes can grow to a very large size as we keep ingesting data. It is necessary to perform a rollover to a new index because there is a limit on how much data can be stored in an index. The rollover index API of Elasticsearch rolls an alias over to a new index when the existing index is considered to be too large. Once the index hits the predefined threshold of 30 GB per shard, a new index with the next rolling number will be created. After rollover, the old index will be available only for fetching data, and the data ingestion happens to the new index.

Master Nodes

Master nodes do not hold any data. They are responsible for managing the cluster state, allocation, creation of indexes and so on.

Hot-Data Node

Hot data nodes perform indexing within the cluster and hold the most recent daily indices that are generally tend to be queried most frequently. Because indexing is a CPU and IO intensive operation, ensure that Hot data nodes are powerful and are backed by attached SSD storage. We recommend running a minimum of 3 Hot nodes for high availability. To increase performance, you may increase this number based on the amount of data you wish to collect and query though.

Warm-Data Node

Warm data nodes are designed to handle a large amount of read-only indices that are not queried frequently. We recommend running a minimum of 3 Hot nodes for high availability. To increase performance, you may increase this number based on the amount of data you wish to collect and query though.

Force Merge

Forcemerge is a process in Elasticsearch that reduces the number of segments in an index. Force Merge service in Elasticsearch Utilities goes over all indexes of CA Jarvis and calls the Elasticsearch force merge API. If warm nodes are present, Forcemerge is always performed on warm nodes. If warm nodes are not present, Forcemerge is performed on the hot nodes itself. Performing Forcemerge on hot nodes can affect the ingestion rate.

Snapshot and Restore

The Snapshot and Restore module of Elasticsearch allows you to create snapshots (like backup) of individual indices or an entire cluster into a file system or into a remote repository like Amazon Simple Storage Service (Amazon S3). Using snapshots, you can create incremental backups, you can restore complete backups or just a few indices.

Installing

- [Docker-Based Installation](#)
- [InstallAnywhere-Based Installation](#)

Docker-Based Installation

- [Docker Swarm Installation](#)
- [Install CA Jarvis Dashboard for CouchDB](#)

Docker Swarm Installation

This article helps you install Docker Swarm for use with CA Jarvis, and CA Jarvis Dashboard.

- [Prerequisites](#)
- [Hardware Requirements](#)
- [Installation Instructions](#)
- [Docker Swam Installation Types](#)
- [Vertical Scaling](#)
- [Docker Volume](#)
- [Clear Docker Swarm](#)

Prerequisites

- Docker Community Edition (CE) 17.09 (swarm initialized)
- Docker Compose 1.16+

Hardware Requirements

Platform	Memory	Processor	Hard Disk
Centos 7.2 to 7.4, 64 bit RHEL 7.2 to 7.4, 64 bit	32 GB	<ul style="list-style-type: none"> ▪ 8-core (for Services layer only) ▪ 16-core (for full stack) 	Hard disk: Minimum of 300-GB space



1. Ensure that your firewall does not block CA Jarvis from the ports it requires. Ensure that the following ports are open on all the machines where you are installing CA Jarvis:
 - Apache Tomcat: **8080** (SSL Disabled) and **8443** (SSL Enabled)
 - Apache Kafka: **9092** and **9093**
 - Apache Zookeeper: **2181**, **2888**, and **3888**

- Elasticsearch: **9300**

2. Ensure that your keyboard and display locale are set to English (UTF-8) only. As a root user, type the following command to see the current locale:

```
locale
```

Installation Instructions

The three installation files that are available in the Docker Swarm Installer can be found [here](#) (2.3) or [here](#) (2.3.1).

Single node production setting yml, vertically scalable by modifying limits

```
jarvis-services.prn.yml
```

Overrides certain variables allowing you to run the installation in a standard Dev environment for testing, POC, demo, and others.

```
jarvis-services.override.dev.yml
```

Overrides certain variables allowing you to run the installation in a three node HA environment.

```
<TODO:jarvis-services.override.ha.yml>
```



Note: Depending on environment, combine configurations before deployment. Run the docker-compose config command to generate a deployable yml file.

Docker Swam Installation Types

The following commands are for the four types of Docker Swarm installations.

Single Node Production Deployment

To install Docker Swarm in a Single Node Production Deployment,

Follow these steps:

1. Verify that Docker Swarm is initialized. If not, use *docker swarm init*.
2. The following labels are required to run the corresponding services:

For Elastic Search

```
jarvis.elasticsearch=true
```

For Zookeeper1:

```
jarvis.zookeeper1=true
```

For Kafka1:

```
jarvis.kafka1=true
```

For APIs, Verifier, Indexer, and Schemaregistry:

```
jarvis.services=true
```

For Utilities and Kron:**Note:** The command may not be required for dev setup.`jarvis.utilities=true`**For CA Jarvis Dashboard:**`jarvis.ldds=true`

For PostgreSQL database to be used with the CA Jarvis Dashboard use:

Note: Not required for external PostgreSQL configuration`jarvis.db=true`

3. To verify the `node_id`, use the command:

`dock node is`

4. To apply the label, use the command:

`docker node update --label-add <label><node_id>`**Example:**`docker node update --label-add jarvis.elasticsearch=true <node_id>`

5. For Docker Compose to configure the services, use the command:

`docker-compose -f jarvis-services.prd.yml config > docker-compose.prd.yml`

6. For Docker Stack to deploy Jarvis, use the command:

`docker stack deploy jarvis --compose-file docker-compose.prd.yml`

Docker Swarm starts the deployment.

7. (Optional) To check the status, run the command:

`docker service ls`

8. (Optional) To get more detailed information about the status, run the command:

`docker stack ps jarvis --no-trunc`

Development Single Node Deployment

To install Docker Swarm for Dev Single Node Deployment,

Follow these steps:

1. Perform steps 1 - 3 from the [Single Node Production Deployment](#) procedure.

2. (Optional) Add the Utilities and Dashboard labels if necessary.

3. For Docker Compose to configure the services, use the command:

`docker-compose -f jarvis-services.prd.yml -f jarvis-services.override.dev.yml config > docker-compose.dev.yml`

4. For Docker Stack to deploy Jarvis, use the command:

`docker stack deploy jarvis --compose-file docker-compose.dev.yml`

Docker Swarm starts the deployment.

5. (Optional) You can use steps 7-8 in the Single Node Deployment to check the deployment status.

Laptop Single Node Deployment

To install Docker Swarm for Laptop Single Node Deployment,

Follow these steps:

1. Perform steps 1 - 3 from the [Single Node Production Deployment](#) procedure.
2. (Optional) Add the Utilities and Dashboard labels if necessary.
3. For Docker Compose to configure the services, use the command:

```
docker-compose -f jarvis-services.prn.yml -f jarvis-services.override.laptop.yml config > docker-compose.laptop.yml
```

4. For Docker Stack to deploy Jarvis, use the command;

```
docker stack deploy jarvis --compose-file docker-compose.laptop.yml
```

Docker Swarm starts the deployment.



Note: Laptop deployment uses tmpfs type for Docker volumes, so data is lost on stop/removal. You can modify the dev override file and can remove tmpfs

Three Node Production High Availability Deployment

To install Docker Swarm for High Availability Deployment,

Follow these steps:

1. Verify that you have three swarm nodes in a cluster, and are acting as manager (one leader).
2. Perform steps 1 - 3 from the [Single Node Production Deployment](#) procedure for the first node.
3. Ensure that you add the labels, *kafka3/zookeeper3* and *kafka3/zookeeper3* for nodes two and three.

Examples:

```
jarvis.zookeeper2=truefor zookeeper2 andjarvis.zookeeper3=truefor zookeeper3.
```

```
jarvis.kafka2=truefor for kafka2 andjarvis.kafka3=truefor kafka3
```



You can only use one set of the kafka/zookeeper label for each node

4. For Docker Compose to configure the services, use the command:

```
docker-compose -f jarvis-services.prn.yml -f jarvis-services.override.ha.yml config > docker-compose.ha.prn.yml
```

5. For Docker Stack to deploy Jarvis, use the command.

```
docker stack deploy jarvis --compose-file docker-compose.ha.prn.yaml
```

Vertical Scaling

Each of the deployment yml comes with pre-set recommended limits for that environment. We may come at a point where we want to increase the capacity vertically because we have hardware available.

By default, the containers take 80 percent of configured container limit as their maximum heap, and half of that for minimum heap. (Elasticsearch takes 60 percent and use same min/max)

Increasing the limit on containers and re-deploy automatically deploys service with new scaled values. For advanced configuration and to control how much ratio we want the process inside the container to take (mostly no need to change), one can use:

RAM_RATIO environment in the container definition inside compose and set it to a value from 0.0-1.0 (the default 0.8 for all except Elasticsearch where its 0.6)

Docker Volume

Named Docker local volume mounting is used by default. You can check the volume using the following command:

```
docker volume ls
```

Use the following command to get more information about the volume:

```
docker volume inspect <volname>
```

Clear Docker Swarm

To use only Docker, remove the stack.

Follow these steps:

1. Use the following command:

```
docker stack rm jarvis
```



Note: The command removes only the services and containers.

2. After all services, tasks, and containers are stopped, use the following command to adjust, or remove, the related volumes:

```
docker volume prune
```

3. (Optional) If you want to remove all stopped containers, unused images, and networks, use the command:

```
docker system prune -a -f
```



Note: The command does not remove volumes.

Install CA Jarvis Dashboard for CouchDB

Prerequisites

- CentOS 7.2 to 7.4

Follow these steps:

1. Download docker-compose.zip file.
2. Copy the .zip file to the target machine. For example, copy the file to the /opt/ folder in the target machine.
3. Run the following command to unzip the file. Ensure that you use the correct name for the .zip file.

```
cd /opt; unzip docker-compose.zip
```

4. Run the following commands one-by-one to install the prerequisites. Docker and Docker compose (1.16.0) are also get installed, if those are not already installed.

```
chmod +x prerequisite.sh
```

```
sh prerequisite.sh
```

5. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi docker-compose.yml
```

6. Provide values for the following variables in the docker-compose.yml file:

```
Under the service named ldds-web:
LDDUSERNAME=COUCHDB_ADMIN_USERNAME
LDDSHOSTNAME=COUCHDB_IPADDRESS
LDDSPORTNO=COUCHDB_PORTNO
LDDUSERPWD=COUCHDB_ADMIN_USERPWD
```

```
Under the service named couchdb:
```

```
IPADDRESS= COUCHDB_IPADDRESS
COUCHDBPORTNO= COUCHDB_PORTNO
```

7. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes that are made to the docker-compose.yml file.

8. Run the following command to bring up the container:

```
docker-compose up -d
```

9. Customize and load the following URL once the Docker container is up to access CA Jarvis Dashboard:

```
http://<<hostname>>:18080
```

InstallAnywhere-Based Installation

- CA Jarvis InstallAnywhere - Single Node

- [CA Jarvis InstallAnywhere - Multi Node](#)
- [Installation of CA Jarvis Dashboard](#)

CA Jarvis InstallAnywhere - Single Node

The following article allows you install both CA Jarvis and CA Jarvis Dashboard using a single script. You must complete the following prerequisite before starting the installation

Prerequisite

1. Install CA Jarvis. Follow one of the following procedures that is based on your requirements, and complete all steps in **Prerequisites** until step 16 in following installation Instructions:
 - [Install CA Jarvis - Non-SSL](#)
 - [Install CA Jarvis - One Way SSL](#)
2. Install CA Dashboard. Follow one of the following procedures that is based on your requirements, and complete all steps in **Prerequisites** until step 9 in following installation Instructions:
 - [Installation of CA Jarvis Dashboard on a Single Node \(Without High-Availability Failover\)](#)
 - [Installation of CA Jarvis Dashboard on a Single Node \(With High-Availability Failover\)](#)

Install CA Jarvis - Non-SSL

The following topic explains the steps to install CA Jarvis on a single node. You can use the same procedure to install Jarvis on the Amazon Web Services (AWS) platform.



A root user must download the installation files and create a non-root sudo user. As a non-root sudo user, you can continue the installation once the root user performs the steps.

We recommend you to install Jarvis as a **non-root sudo user** so that Jarvis is installed as a service. You can use the systemctl commands to start and stop the services.

If you are installing Jarvis as a non-root non-sudo user, Jarvis is installed as an application. You must use shell scripts to start and stop the components. You must follow the random steps that are marked for non-root non-sudo user to complete the installation as a non-root non-sudo user.

Prerequisites

Software and Hardware Requirements

Platform	Memory	Processor	Hard Disk	Others
	32 GB	8-Core		

Platform	Memory	Processor	Hard Disk	Others
<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above. 			Hard disk: Minimum of 300 GB space	<ul style="list-style-type: none"> ▪ Oracle Java 1.8.0_131 (JDK or JRE). Use the command <code>java -version</code> to know the version of Java. Refer Troubleshooting, for information on Java configurations ▪ Unzip utility. ▪ Dos2Unix utility. ▪ Netcat utility. ▪ Bind-utils (Only for configuring SSL)

Other Requirements

- Ensure that the `/etc/hosts` in all your servers have the correct value set in the following format:

<IP Address> <Host Name with domain> <Alias>

Example:

10.131.58.124 user05-I193354.ca.com user05-I193354

For non-root and non-sudo Users

- Ensure that `ulimit` is set to 65536 and `vm.max_map_count` is set to 262144 for Elasticsearch to start successfully by completing the following procedures:

Do the following steps as a root user to set ulimit

1.
 - a.
 - i. Change to the `/etc/security` directory.
 - ii. Add the following lines to the `limits.conf` file:
 - * ulimit -n 65536
 - * hard nproc 65536
 - * soft nproc 65536
 - * soft nofile 65536
 - * hard nofile 131072

Do the following steps as a root user to set vm.max_map_count:

1.
 - a. Change to the `/etc` directory.
 - b. Update the `sysctl.conf` file with the following value:

```
vm.max_map_count=262144
```
 - c. Run the following command to get the changes reflected without restarting your node:

```
sysctl -q -w vm.max_map_count=262144
```

- Ensure that your firewall does not block CA Jarvis from the ports it requires. Ensure that the following ports are open on all the machines where you are installing CA Jarvis:
 - Apache Tomcat: **8080** (SSL Disabled) and **8443** (SSL Enabled)
 - Apache Kafka: **9092** and **9093**

- Apache Zookeeper: **2181**, **2888**, and **3888**

- Elasticsearch: **9300**

Run the following commands to list the open ports:

You can skip the first command if Network Mapper (nmap) is already installed.

```
yum install nmap
```

```
nmap localhost
```



Ensure that your keyboard and display locale are set to English (UTF-8) only. As a root user, type the following command to see the current locale:

```
locale
```

Installation Instructions

The following procedure explains the steps to install Jarvis on a single node.



Note: You can find the errors that might happen during the installation, and the possible resolutions are located [here](#).

Refer [Troubleshooting](#), if you encounter issues during the installation.

In this procedure, the term "ports" indicates Transport Control Protocol (TCP) ports.

1. Log in as a root user.
2. Download latest version of the Jarvis installer package (.tar file) from CA Support site.
3. Copy the .tar file to the target machine. For example, copy the file to the */tmp* folder in the target machine.
4. Run the following command to untar the file. Ensure that you use the correct name for the .tar file.

```
cd /tmp; tar -xvf jarvisInstaller_installAnywhere-<version>.tar
```

This command creates */<location of the tar file>/jarvisInstaller_installAnywhere*, which contains the installation files and Jarvis binaries.

5. Ignore step 5 and 6, if you have already created a non-root user, and will use the same user for installing Jarvis. If want to create a non-root user for installing Jarvis, run the following command and continue with step 6:

```
cd jarvisInstaller_installAnywhere/scripts
```

6. Create a **non-root sudo user** by running the following commands. From step 10, you are using the non-root-user credentials that you have set in this step.

```
chmod +x prepareMachineAsRoot.sh
./prepareMachineAsRoot.sh <username>
```

<username> must be a string.

For non-root and non-sudo Users

Create a **non-root non-sudo user** by running the following command. Ensure that you customize the command and provide password for the non-root user when prompted:

```
useradd <username>
```

<username> must be a string.

7. Run the following command:

```
cd /tmp
```

8. Run the following commands one by one:

```
chown -R <username>:<username> jarvisInstaller_installAnywhere
```

9. Ensure that <username> has the read/write permissions to the directories to be used during the installation. Required if you are using a directory outside of <username>'s home directory to set values for USER_INSTALL_DIR, KAFKA_MOUNT, or ES_DATA_MOUNT. To set permissions, complete the following steps:

- a. Change to the directory to which you want to set the permissions.

- b. Customize and run the following commands:

```
chmod 755 <name of the directory>
```

```
chown -R <username>:<username> <name of the directory>
```

10. Log in as the non-root user to whom the permissions were set in step 6.

```
su - <username>
```

11. Copy the jarvisInstaller_installAnywhere directory to the home directory of <username> (/home/<username>):

Example:

```
cp -r /tmp/jarvisInstaller_installAnywhere /home/<username>/.
```

12. Run the following command:

```
cd jarvisInstaller_installAnywhere/properties
```

13. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi analyticsInstaller.properties
```

14. Make the following changes in the **analyticsInstaller.properties** file:



Because this file contains passwords, ensure that you have elevated permissions for this file.

If you are editing a commented value, for example, `#INSTALL_API=true`, ensure that you uncomment (remove #) while editing.

Ensure not to include any additional spaces after the values entered in the **analyticsInstaller.properties** file. For example, there should not be any spaces after the Value Y when you set the following parameter:

`SINGLE_MACHINE=Y.`

a. Provide the following information:

- i. **USER_INSTALL_DIR:** Provide the location where Jarvis should be installed. Ensure that <username> has read/write permissions to this directory as mentioned in step 9.

Example:

```
### Installation Details ###
USER_INSTALL_DIR=/home/<username>/Analytics
SINGLE_MACHINE=Y
SSL_ENABLED=false
```

b. Do not change anything, if you want to install all the services of Jarvis. Uncomment (remove #) and change the value to **false**, should you want to customize the installation:

```
#INSTALL_API=true
#INSTALL_VERIFIER=true
#INSTALL_KAFKA=true
#INSTALL_INDEXER=true
#INSTALL_ELASTICSEARCH=true
#INSTALL_KRON=true
#INSTALL_ES_UTILS=true
#INSTALL_SCHEMA_REGISTRY=true
```

c. Uncomment (remove #) the following line and provide absolute path to the mounted data directory, should you want to change the default location. Ensure that <username> has read/write permissions to the mounted data directory as mentioned in step 9.

```
#KAFKA_MOUNT=$USER_INSTALL_DIR$/Data/kafkaMNT
#ZOOKEEPER_MOUNT=$USER_INSTALL_DIR$/Data/zkMNT
Example:
#KAFKA_MOUNT=/home/<username>/Analytics/Data/kafkaMNT
#ZOOKEEPER_MOUNT=/home/<username>/Analytics/Data/zkMNT
```

d. Provide the following details:

- i. **ES_CLUSTER_NAME:** Provide a name for the ElasticSearch cluster.
- ii. **ES_DATA_MOUNT:** Provide absolute path to the mounted data directory. Ensure that <username> has read/write permissions to the mounted data directory as mentioned in step 9.



During an upgrade, the data mount directory may get removed. Ensure that you provide a path outside the Elasticsearch directory as the value for ES_DATA_MOUNT.

Example:

```
#ES_DATA_MOUNT=/home/<username>/Analytics/Data/es
```

- iii. **TOMCAT_INSTALLATION_HOST**=host1:port1: Provide the IP address or FQDN and port number (8080) of the host where Tomcat should be installed.
 - iv. **VERIFIER_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Verifier should be installed.
 - v. **INDEXER_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Indexer should be installed.
 - vi. **ESUTILS_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Elasticsearch utilities should be installed.
 - vii. **KRON_HOST**=localhost:8080: Provide the IP address or FQDN of the host where KRON should be installed.
- e. (Optional) Update the following variables to define the total number of Kafka consumers in Verifier and Indexer. PRIORITY_RATIO is calculated based on these values. For example, if the count is defined as 100, value of P1 is 80, value of P2 is 10, and value of P3 is 10.
- ```
Total no of kafka consumer in Verifier
VERIFIER_KAFKA_CONSUMER_COUNT=4
Total no of kafka consumer in Indexer
INDEXER_KAFKA_CONSUMER_COUNT=4
```
- f. (Optional) Uncomment the following line to define the priorities for data ingestion.
- ```
#Priority ratio
#PRIORITY_RATIO=P1:80,P2:10,P3:10
```

**Example:**

```
PRIORITY_RATIO= P1:70,P2:20,P3:10
```

In this example, **P1**, **P2**, and **P3** are the priorities for the messages to be processed. You can define up to nine levels of priorities. 70,20, and 10 are the percentage of resources that must be allocated to each priority bucket (topics in kafka) for processing. The sum of the percentages should not exceed 100.

If a machine or container has 4 cores, then there are 8 threads /consumers that will be divided proportionately according to the priority ratio.

```
PRIORITY_RATIO= P1:70,P2:20,P3:10 and number of cores = 4.
```

There are 8 threads/consumers in the resource pool. So, approximately P1 gets 6 consumers , P2 gets 1 consumer, and P3 gets 1 consumer.

While sending the data (ingestion), you can provide the priority (**priority**) in the HTTP header of the input messages. Example: `curl -XPOST -H 'priority: p3' <ingestion url>`

If **PRIORITY_RATIO** is not set, then by default all the data will be handled by P1 priority and all the resources will be allocated to it.

g. (Optional) Update the following details:

```
#ES_SNAPSHOT location
#ES_SNAPSHOT_LOCATION=/opt/CA/Analytics/Data/backup : The location
where Elasticsearch stores snapshot.
#Exclude indices from snapshot
#ES_SNAPSHOT_EXCLUDE_INDICES=index_list : The list of indexes that
should be excluded during back up.
#Datapurge retention period.
#DATAPURGE_DEFAULT_RETENTION_PERIOD=45 : The number of days up to
which the data should be kept.
#ES_SNAPSHOT_REPOSITORY_NAME=RepositoryName: Before any snapshot or
restore operation can be performed, a snapshot repository should be
registered in Elasticsearch. This variable indicates the name of such
a repository.
#INDEX_LIMIT:NUMBER_OF_SHARDS_PER_NODE: Assign the number of shards
that will be present on each node of the Elasticsearch cluster.
By restricting the number of shards allocated to each node, you can
distribute the load across the Elasticsearch cluster.
This value is directly dependent on the following values:
• Number of available nodes.
• Number of primary shards present for the index.
If the value is very low, it affects the allocation of the primary
shards. If the value is very high, it degrades the performance of the
node.
If you do not assign any value for this parameter, the system
calculates the value based on the number of available nodes and
primary shards. Even if you have assigned a value for this parameter,
the system checks the assigned value against the value calculated by
the system. It then uses the value that suits better for your cluster.
Example:
Your setup is as follows:
• Number of available nodes = 3
• Number of primary shards present for the index = 5 (Replication
factor=1)
The system determines that the suitable shards per node is 5. By
doing so, the cluster will be safe even in the case of a node failure.
In this example, using a very low value like 1 (primary shards
themselves will be unassigned) or a very high value like 10 (results
in an imbalance within the cluster) is not recommended.
```

15. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes that are made to the `analyticsInstaller.properties` file.

16. Change to the parent directory :

Example:

```
cd /home/<username>/jarvisInstaller_installAnywhere
```

17. Run the following commands to install Jarvis:

```
./CA_Analytics.bin -f <path>/analyticsInstaller.properties
```

Example:

```
./CA_Analytics.bin -f properties/analyticsInstaller.properties
```


The installer displays the progress of installation. The Jarvis Health-check is executed as part of the installation on a single node. The installer displays the location of the log file to see the progress of the execution of Jarvis Health-check.

Jarvis is installed regarding the configurations in the **analyticsInstaller.properties** file. The On successful installation, the installer starts all the installed services.



If you get a message saying that port is already in use, change the port number in the analyticsInstaller.properties file and try to install again.

18. **(Optional)** Go to the following directory to see the log files that are generated by the installer:

`<USER_INSTALL_DIR>/_CA Analytics_installation/Logs`

19. **(Optional:)** Execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Command Format for starting and stopping services:

`sudo systemctl <command><serviceName>`

Command Format for checking the status of services:

`systemctl <command> <serviceName>`

Where **<command>** can be one of the following:

start

stop

status

And, **<serviceName>** can be one of the following:

ca_zookeeper.service should be started before starting **ca_kafka.service**.

After starting **ca_kafka.service**, the rest of the services can be started in any order.

ca_zookeeper.service
ca_kafka.service
ca_elasticsearch.service
ca_schema-registry.service
ca_tomcat.service
ca_verifier-ingest.service
ca_indexer-ingest.service
ca_esutils.service

Applies for non-root non-sudo user) Change to <USER_INSTALL_DIR>/bin and execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Applies for non-root non-sudo user) Change to <USER_INSTALL_DIR>/bin and execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Command for starting the Jarvis services:

`./startElasticsearch.sh`
`./startZookeeper.sh`
`./startkafka.sh`
`./startSchemaRegistry.sh`
`./startIndexerIngest.sh`
`./startIndexerUpdate.sh`

```
./startVerifierIngest.sh
./startVerifierUpdate.sh
./startESUtils.sh
./startTomcat.sh
```

Command for stopping the Jarvis services:

```
./stopElasticsearch.sh
./stopSchemaRegistry.sh
./stopZookeeper.sh
./stopkafka.sh
./stopIndexerIngest.sh
./stopIndexerUpdate.sh
./stopVerifierIngest.sh
./stopVerifierUpdate.sh
./stopESUtils.sh
./stopTomcat.sh
```

Installed Jarvis. What Next?

You are now ready to start [using Jarvis APIs](#).

Install CA Jarvis - One Way SSL

The following topic explains the steps to install CA Jarvis on a single node with SSL enabled. You can use the same procedure to install Jarvis on Amazon Web Services (AWS) platform.



A root user must download the installation files and create a non-root sudo user. As a non-root sudo user, you can continue the installation once the root user performs his steps.

We recommend you to install Jarvis as a **non-root sudo user** so that Jarvis will be installed as a service. You can use the systemctl commands to start and stop the services.

If you are installing Jarvis as a non-root non-sudo user, Jarvis will be installed as an application. You must use shell scripts to start and stop the components. You must follow the random steps marked for non-root non-sudo user to complete the installation as a non-root non-sudo user.

Prerequisites

Software and Hardware Requirements

Platform	Memory	Processor	Hard Disk	Others
<ul style="list-style-type: none"> RHEL 7.2 CentOS 7.2 or above. 	32 GB	8-Core	Hard disk: Minimum of 300 GB space	<ul style="list-style-type: none"> Oracle Java 1.8.0_131 (JDK or JRE). Use the command <code>java -version</code> to know the version of Java. Refer Troubleshooting, for information on Java configurations Unzip utility. Dos2Unix utility. Netcat utility. Bind-utils (Only for configuring SSL)

Secure Socket Layer (SSL) Requirements

For securing the communication channel among various components using Secure Socket Layer (SSL) authentication, [set up the SSL certificates](#). Note down the the following values after setting up the SSL certificates:

- Location where keystore.jks file and truststore.jks file are stored
- The keystore and truststore passwords
- Fully-Qualified Domain Name (FQDN)

If you are installing Jarvis on multi nodes, repeat the steps on each node. So, before the installation, you are just setting up the certificates and after the installation, you can enable SSL.

Other Requirements

- Ensure that the **/etc/hosts** in all your servers have the correct value set in the following format:

<IP Address> <Host Name with domain> <Alias>

Example:

10.131.58.124 user05-I193354.ca.com user05-I193354

For non-root and non-sudo Users

- ▪ Ensure that *ulimit* is set to 65536 and *vm.max_map_count* is set to 262144 for Elasticsearch to start successfully by completing the following procedures:

Do the following steps as a root user to set ulimit

1. a. i. Change to the */etc/security* directory.
- ii. Add the following lines to the *limits.conf* file:


```
* ulimit -n 65536
* hard nproc 65536
* soft nproc 65536
* soft nofile 65536
* hard nofile 131072
```

Do the following steps as a root user to set vm.max_map_count:

1. a. Change to the */etc* directory.
- b. Update the *sysctl.conf* file with the following value:


```
vm.max_map_count=262144
```
- c. Run the following command to get the changes reflected without restarting your node:


```
sysctl -q -w vm.max_map_count=262144
```

- Ensure that your firewall does not block CA Jarvis from the ports it requires. Ensure that the following ports are open on all the machines where you are installing CA Jarvis:

- Apache Tomcat: **8080** (SSL Disabled) and **8443** (SSL Enabled)
- Apache Kafka: **9092** and **9093**
- Apache Zookeeper: **2181**, **2888**, and **3888**

- Elasticsearch: **9300**

Run the following commands to list the open ports:

You can skip the first command if Network Mapper (nmap) is already installed.

```
yum install nmap
```

```
nmap localhost
```



Ensure that your keyboard and display locale are set to English (UTF-8) only. As a root user, type the following command to see the current locale:

```
locale
```

Installation Instructions

The following procedure explains the steps to install Jarvis on a single node with SSL enabled.



You can find the errors that might happen during the installation, and the possible resolutions [here](#).

Refer [Troubleshooting](#), if you encounter issues during the installation.

In this procedure, the term "ports" indicates Transport Control Protocol (TCP) ports.

1. Log in as a root user.
2. Download latest version of the Jarvis installer package (.tar file) from CA Support site.
3. Copy the .tar file to the target machine. For example, copy the file to the */tmp* folder in the target machine.
4. Run the following command to untar the file. Ensure that you use the correct name for the .tar file.

```
cd /tmp; tar -xvf jarvisInstaller_installAnywhere-<version>.tar
```

This command creates */<location of the tar file>/jarvisInstaller_installAnywhere*, which contains the installation files and Jarvis binaries.

5. Ignore step 5 and 6, if you have already created a non-root user, and want to use the same user for installing Jarvis. If want to create a non-root user for installing Jarvis, run the following command and continue with step 6:

```
cd jarvisInstaller_installAnywhere/scripts
```

6. Create a **non-root sudo user** by running the following commands. From step 10, you will be using the non-root-user credentials that you have set in this step.

```
chmod +x prepareMachineAsRoot.sh
./prepareMachineAsRoot.sh <username>
```

<username> must be a string.

For non-root and non-sudo Users

Create a **non-root non-sudo user** by running the following command. Ensure that you customize the command and provide password for the non-root user when prompted:

```
useradd <username>
```

<username> must be a string.

7. Run the following command:

```
cd /tmp
```

8. Run the following commands one by one:

```
chown -R <username>:<username> jarvisInstaller_installAnywhere
```

```
chmod +x jarvisInstaller_installAnywhere/CA_Analytics.bin
```

9. Ensure that <username> has the read/write permissions to the directories that are going to be used during the installation. This is required if you are using a directory outside of <username>'s home directory to set values for USER_INSTALL_DIR, KAFKA_MOUNT, or ES_DATA_MOUNT. To set permissions, complete the following steps:

a. Change to the directory to which you want to set the permissions.

b. Customize and run the following commands:

```
chmod 755 <name of the directory>
```

```
chown -R <username>:<username> <name of the directory>
```

10. Log in as the non-root user to whom the permissions were set in step 6.

```
su - <username>
```

11. Copy the jarvisInstaller_installAnywhere directory to the home directory of <username> (/home/<username>) :

Example:

```
cp -r /tmp/jarvisInstaller_installAnywhere /home/<username>/.
```

12. Run the following command:

```
cd jarvisInstaller_installAnywhere/properties
```

13. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi analyticsInstaller.properties
```

14. Make the following changes in the **analyticsInstaller.properties** file:



Because this file contains passwords, ensure that you have elevated permissions for this file.

If you are editing a commented value, for example, `#INSTALL_APIS=true`, ensure that you uncomment (remove #) while editing.

Ensure that there are no additional spaces after the values entered in the **analyticsInstaller.properties** file. For example, there should not be any spaces after the Value Y when you set the following parameter:

```
SINGLE_MACHINE=Y.
```

a. Provide the following information:

- i. **USER_INSTALL_DIR**: Provide the location where Jarvis should be installed. Ensure that <username> has read/write permissions to this directory as mentioned in step 9.
- ii. **SSL_ENABLED**: Change to true to enable SSL. (Two-way SSL for Elasticsearch. One-way SSL for Kafka and Tomcat services)
- iii. **SSL_TWO_WAY_ENABLED**: Change to **true**, if you want to enable two-way SSL for Tomcat services.

Example:

```
### Installation Details ###
USER_INSTALL_DIR=/home/<username>/Analytics
SINGLE_MACHINE=Y
SSL_ENABLED=true
SSL_TWO_WAY_ENABLED:true
```

b. Set the following SSL-related values:



Ensure that the user who is installing CA Jarvis has read permission to the location where the Keystore and Truststore files reside.

KEYSTORE_FILEPATH: Path where the keystore.jks file is located.

TRUSTSTORE_FILEPATH: Path where the truststore.jks file is located.

KEYSTORE_PASSWORD: The password that is used to access the keystore file when SSL is enabled. Use the same password that you used while you have created the certificates.

TRUSTSTORE_PASSWORD: The password that is used to access the truststore file

when SSL is enabled. Use the same password that you used while you have created the certificates.

KEY_PASSWORD: The password that is used to access the key when SSL is enabled. Use the same password that you used while you have created the certificates.

Example:

```
KEYSTORE_FILEPATH=/home/qa/jarvis/sslCerts/CN=ciavrov-m-keystore.jks
TRUSTSTORE_FILEPATH=/home/qa/jarvis/sslCerts/truststore.jks
KEYSTORE_PASSWORD=1237c3b0ef108fe28b90
TRUSTSTORE_PASSWORD=dc075fba9dd181b77fb1
KEY_PASSWORD=abcd1234
```

- c. Do not change anything, if you want to install all the services of Jarvis. Uncomment (remove #) and change the value to **false**, if you want to customize the installation:

```
#INSTALL_API=true
#INSTALL_VERIFIER=true
#INSTALL_KAFKA=true
#INSTALL_INDEXER=true
#INSTALL_ELASTICSEARCH=true
#INSTALL_KRON=true
#INSTALL_ES_UTILS=true
#INSTALL_SCHEMA_REGISTRY=true
```

- d. Uncomment (remove #) the following line and provide absolute path to the mounted data directory, if you want to change the default location. Ensure that <username> has read/write permissions to the mounted data directory as mentioned in step 9.

```
#KAFKA_MOUNT=$USER_INSTALL_DIR$/Data/kafkaMNT
#ZOOKEEPER_MOUNT=$USER_INSTALL_DIR$/Data/zkMNT
Example:
#KAFKA_MOUNT=/home/<username>/Analytics/Data/kafkaMNT
#ZOOKEEPER_MOUNT=/home/<username>/Analytics/Data/zkMNT
```

- e. Provide the following details:

- i. **ES_CLUSTER_NAME:** Provide a name for the Elasticsearch cluster.
- ii. **ES_DATA_MOUNT:** Provide absolute path to the mounted data directory. Ensure that <username> has read/write permissions to the mounted data directory as mentioned in step 9.



During an upgrade, the data mount directory may get removed. Ensure that you provide a path outside the Elasticsearch directory as the value for ES_DATA_MOUNT.

Example:

```
#ES_DATA_MOUNT=/home/<username>/Analytics/Data/es
```

- iii. **TOMCAT_INSTALLATION_HOST=host1:port1:** Provide the IP address or FQDN and port number (8443) of the host where Tomcat should be installed.
- iv. **VERIFIER_INSTALLATION_HOSTS=host1,host2:** Provide the IP address or FQDN of the host where Verifier should be installed.

- v. **INDEXER_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Indexer should be installed.
 - vi. **ESUTILS_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Elasticsearch utilities should be installed.
 - vii. **KRON_HOST**=localhost:8443: Provide the IP address or FQDN of the host where KRON should be installed.
- f. (Optional) Update the following variables to define the total number of Kafka consumers in Verifier and Indexer. **PRIORITY_RATIO** is calculated based on these values. For example, if the count is defined as 100, value of P1 will be 80, value of P2 will be 10, and value of P3 will be 10.

```
### Total no of kafka consumer in Verifier
VERIFIER_KAFKA_CONSUMER_COUNT=4
### Total no of kafka consumer in Indexer
INDEXER_KAFKA_CONSUMER_COUNT=4
```

- g. (Optional) Uncomment the following line to define the priorities for data ingestion.

```
#Priority ratio
#PRIORITY_RATIO=P1:80,P2:10,P3:10
```



Example:

PRIORITY_RATIO= P1:70,P2:20,P3:10

In this example, **P1**, **P2**, and **P3** are the priorities for the messages to be processed. You can define up to nine levels of priorities. 70,20, and 10 are the percentage of resources that must be allocated to each priority bucket (topics in kafka) for processing. The sum of the percentages should not exceed 100.

If a machine or container has 4 cores, then there are 8 threads /consumers that will be divided proportionately according to the priority ratio.

PRIORITY_RATIO= P1:70,P2:20,P3:10 and number of cores = 4.

There are 8 threads/consumers in the resource pool. So, approximately P1 gets 6 consumers , P2 gets 1 consumer, and P3 gets 1 consumer.

While sending the data (ingestion), you can provide the priority (**priority**) in the HTTP header of the input messages. Example: curl -XPOST -H '**priority: p3**' <ingestion url>

If **PRIORITY_RATIO** is not set, then by default all the data will be handled by P1 priority and all the resources will be allocated to it.

- h. (Optional) Update the following details:

```
#ES SNAPSHOT location
#ES_SNAPSHOT_LOCATION=/opt/CA/Analytics/Data/backup : The location
where Elasticsearch stores snapshot.
#Exclude indices from snapshot
#ES_SNAPSHOT_EXCLUDE_INDICES=index_list : The list of indexes that
```


should be excluded during back up.
#Datapurge retention period.
#DATAPURGE_DEFAULT_RETENTION_PERIOD=45 : The number of days up to which the data should be kept.
#ES_SNAPSHOT_REPOSITORY_NAME=RepositoryName: Before any snapshot or restore operation can be performed, a snapshot repository should be registered in Elasticsearch. This variable indicates the name of such a repository.
#ES_SNAPSHOT_REPOSITORY_NAME=RepositoryName: Before any snapshot or restore operation can be performed, a snapshot repository should be registered in Elasticsearch. This variable indicates the name of such a repository.
#INDEX_LIMIT:NUMBER_OF_SHARDS_PER_NODE: Assign the number of shards that will be present on each node of the Elasticsearch cluster. By restricting the number of shards allocated to each node, you can distribute the load across the Elasticsearch cluster. This value is directly dependent on the following values:

- Number of available nodes.
- Number of primary shards present for the index.

If the value is very low, it affects the allocation of the primary shards. If the value is very high, it degrades the performance of the node.
 If you do not assign any value for this parameter, the system calculates the value based on the number of available nodes and primary shards. Even if you have assigned a value for this parameter, the system checks the assigned value against the value calculated by the system. It then uses the value that suits better for your cluster.
Example:
 Your setup is as follows:

- Number of available nodes = 3
- Number of primary shards present for the index = 5 (Replication factor=1)

The system determines that the suitable shards per node is 5. By doing so, the cluster will be safe even in the case of a node failure. In this example, using a very low value like 1 (primary shards themselves will be unassigned) or a very high value like 10 (results in an imbalance within the cluster) is not recommended.

i. Provide the following details if your CA Jarvis Dashboard setup is SSL-enabled:

i. **LDDS_SSL_CONFIGURED=false**. Change to **true** if CA Jarvis Dashboard is SSL-enabled.

ii. **LDDS_HOST=localhost:8080**. Provide the IP address or FQDN of the CA Jarvis Dashboard host.

15. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes made to the `analyticsInstaller.properties` file.

16. Change to the parent directory :

Example:

```
cd /home/<username>/jarvisInstaller_installAnyWhere
```

17. Run the following commands to install Jarvis:

```
./CA_Analytics.bin -f <path>/analyticsInstaller.properties
```

Example:

```
./CA_Analytics.bin -f properties/analyticsInstaller.properties
```

The installer displays the progress of installation. The Jarvis Health-check is executed as part of the installation on a single node. The installer displays the location of the log file to see the progress of the execution of Jarvis Health-check.

Jarvis is installed as per the configurations in the **analyticsInstaller.properties** file. The On successful installation, the installer starts all the installed services.



If you get a message saying that port is already in use, change the port number in the `analyticsInstaller.properties` file and try to install again.

18. **(Optional)** Go to the following directory to see the log files generated by the installer:

`<USER_INSTALL_DIR>/_CA Analytics_installation/Logs`

19. **(Optional:)** Execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Command Format for starting and stopping services:

`sudo systemctl <command><serviceName>`

Command Format for checking the status of services:

`systemctl <command> <serviceName>`

Where `<command>` can be one of the following:

start

stop

status

And, `<serviceName>` can be one of the following:

ca_zookeeper.service should be started before starting **ca_kafka.service**.

After starting **ca_kafka.service**, the rest of the services can be started in any order.

ca_zookeeper.service
ca_kafka.service
ca_elasticsearch.service
ca_schema-registry.service
ca_tomcat.service
ca_verifier-ingest.service
ca_indexer-ingest.service
ca_esutils.service

Applies for non-root non-sudo user) Change to `<USER_INSTALL_DIR>/bin` and execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Applies for non-root non-sudo user) Change to `<USER_INSTALL_DIR>/bin` and execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Command for starting the Jarvis services:

```
./startElasticsearch.sh
./startZookeeper.sh
./startkafka.sh
./startSchemaRegistry.sh
./startIndexerIngest.sh
./startIndexerUpdate.sh
./startVerifierIngest.sh
./startVerifierUpdate.sh
./startESUtils.sh
./startTomcat.sh
```

Command for stopping the Jarvis services:

```
./stopElasticsearch.sh
```

```
./stopSchemaRegistry.sh
./stopZookeeper.sh
./stopkafka.sh
./stopIndexerIngest.sh
./stopIndexerUpdate.sh
./stopVerifierIngest.sh
./stopVerifierUpdate.sh
./stopESUtils.sh
./stopTomcat.sh
```

Installed Jarvis. What Next?

You are now ready to start [using Jarvis APIs](#).

CA Jarvis InstallAnywhere - Multi Node

The following article allows you install CA Jarvis Multi Node using a single script. You must complete the following prerequisite before starting the installation

Prerequisite

1. Install CA Jarvis. Follow one of the following procedures that is based on your requirements:

- [CA Jarvis - Non SSL](#)
- [CA Jarvis - One Way SSL](#)

CA Jarvis IA - One Way SSL

The following topic explains the steps to install CA Jarvis on multiple nodes with SSL enabled.



A root user must download the installation files and create a non-root sudo user. As a non-root sudo user, you can continue the installation once the root user performs his steps.

We recommend you to install Jarvis as a **non-root sudo user** so that Jarvis will be installed as a service. You can use the systemctl commands to start and stop the services.

If you are installing Jarvis as a non-root non-sudo user, Jarvis will be installed as an application. You must use shell scripts to start and stop the components. You must follow the random steps marked for non-root non-sudo user to complete the installation as a non-root non-sudo user.

Prerequisites

Software and Hardware Requirements



The Minimum values mentioned here are for development and test environments, and the Recommended values are for production environment.

Component	Platform	Memory		Processor		Hard Disk	
		Minimum	Recommended	Minimum	Recommended	Minimum	Recommended
Kafka & Zookeeper	<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above. 	8 GB	32 GB	2-Core	4-Core	100 GB Solid State Device (SSD)	<ul style="list-style-type: none"> ▪ 800 GB SSD
Elasticsearch	<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above. 	8 GB	64 GB	4-Core	8-Core	100 GB SSD	800 GB SSD with RAID0 format
Verifier	<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above. 	4 GB	8 GB	2-Core	4-Core	100 GB SSD	100 GB SSD

Component	Platform	Memory		Processor		Hard Disk	
-----------	----------	--------	--	-----------	--	-----------	--

Indexer	▪ RHEL 7.2	4 GB	8 GB	2-Core	4-Core	100 GB SSD	100 GB SSD
	▪ CentOS 7.2 or above.						

Apache Tomcat	▪ RHEL 7.2	4 GB	8 GB	2-Core	4-Core	100 GB SSD	100 GB SSD
	▪ CentOS 7.2 or above.						

Secure Socket Layer (SSL) Requirements

For securing the communication channel among various components using Secure Socket Layer (SSL) authentication, [set up the SSL certificates](#). Note down the the following values after setting up the SSL certificates:

- Location where keystore.jks file and truststore.jks file are stored
- The keystore and truststore passwords
- Fully-Qualified Domain Name (FQDN)

If you are installing Jarvis on multi nodes, repeat the steps on each node. So, before the installation, you are just setting up the certificates and after the installation, you can enable SSL.

Other Requirements

- Ensure that the **/etc/hosts** in all your servers have the correct value set in the following format:

<IP Address> <Host Name with domain> <Alias>

Example:

10.131.58.124 user05-I193354.ca.com user05-I193354

For non-root and non-sudo Users

- Ensure that *ulimit* is set to 65536 and *vm.max_map_count* is set to 262144 for Elasticsearch to start successfully by completing the following procedures:

Do the following steps as a root user to set ulimit

1. a. i. Change to the */etc/security* directory.
 - ii. Add the following lines to the *limits.conf* file:


```
* ulimit -n 65536
* hard nproc 65536
* soft nproc 65536
* soft nofile 65536
* hard nofile 131072
```

Do the following steps as a root user to set vm.max_map_count:

1. a. Change to the */etc* directory.
 - b. Update the *sysctl.conf* file with the following value:


```
vm.max_map_count=262144
```
 - c. Run the following command to get the changes reflected without restarting your node:


```
sysctl -q -w vm.max_map_count=262144
```

- Ensure that your firewall does not block CA Jarvis from the ports it requires. Ensure that the following ports are open on all the machines where you are installing CA Jarvis:

- Apache Tomcat: **8080** (SSL Disabled) and **8443** (SSL Enabled)
- Apache Kafka: **9092** and **9093**
- Apache Zookeeper: **2181**, **2888**, and **3888**

- Elasticsearch: **9300**

Run the following commands to list the open ports:

You can skip the first command if Network Mapper (nmap) is already installed.

```
yum install nmap
```

```
nmap localhost
```



Ensure that your keyboard and display locale are set to English (UTF-8) only. As a root user, type the following command to see the current locale:

```
locale
```

Installation Instructions

The following procedure explains the steps to install Jarvis on multiple nodes with SSL enabled.



You can find the errors that might happen during the installation, and the possible resolutions [here](#).

Refer [Troubleshooting](#), if you encounter issues during the installation.

In this procedure, the term "ports" indicates Transport Control Protocol (TCP) ports.

1. Log in as a root user.
2. Download latest version of the Jarvis installer package (.tar file) from CA Support site.
3. Copy the .tar file to the target machine. For example, copy the file to the `/tmp` folder in the target machine.
4. Run the following command to untar the file. Ensure that you use the correct name for the .tar file.

```
cd /tmp; tar -xvf jarvisInstaller_installAnywhere-<version>.tar
```

This command creates `/<location of the tar file>/jarvisInstaller_installAnywhere`, which contains the installation files and Jarvis binaries.

5. Ignore step 5 and 6, if you have already created a non-root user, and want to use the same user for installing Jarvis. If want to create a non-root user for installing Jarvis, run the following command and continue with step 6:

```
cd jarvisInstaller_installAnywhere/scripts
```

6. Create a **non-root sudo user** by running the following commands. From step 10, you will be using the non-root-user credentials that you have set in this step.

```
chmod +x prepareMachineAsRoot.sh
./prepareMachineAsRoot.sh <username>
```

`<username>` must be a string.

For non-root and non-sudo Users

Create a **non-root non-sudo user** by running the following command. Ensure that you customize the command and provide password for the non-root user when prompted:

```
useradd <username>
```

<username> must be a string.

7. Run the following command:

```
cd /tmp
```

8. Run the following commands one by one:

```
chown -R <username>:<username> jarvisInstaller_installAnywhere
```

```
chmod +x jarvisInstaller_installAnywhere/CA_Analytics.bin
```

9. Ensure that <username> has the read/write permissions to the directories that are going to be used during the installation. This is required if you are using a directory outside of <username>'s home directory to set values for USER_INSTALL_DIR, KAFKA_MOUNT, or ES_DATA_MOUNT. To set permissions, complete the following steps:

- a. Change to the directory to which you want to set the permissions.

- b. Customize and run the following commands:

```
chmod 755 <name of the directory>
```

```
chown -R <username>:<username> <name of the directory>
```

10. Log in as the non-root user to whom the permissions were set in step 6.

```
su - <username>
```

11. Copy the jarvisInstaller_installAnywhere directory to the home directory of <username> (/home/<username>) :

Example:

```
cp -r /tmp/jarvisInstaller_installAnywhere /home/<username>/.
```

12. Run the following command:

```
cd jarvisInstaller_installAnywhere/properties
```

13. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi analyticsInstaller.properties
```

14. Make the following changes in the **analyticsInstaller.properties** file:



Because this file contains passwords, ensure that you have elevated permissions for this file.

If you are editing a commented value, for example, `#INSTALL_API=true`, ensure that you uncomment (remove #) while editing.

Ensure that there are no additional spaces after the values entered in the **analyticsInstaller.properties** file. For example, there should not be any spaces after the Value Y when you set the following parameter:

`SINGLE_MACHINE=Y.`

a. Provide the following information:

- i. **USER_INSTALL_DIR:** Provide the location where Jarvis should be installed. Ensure that <username> has read/write permissions to this directory as mentioned in step 9.
- ii. **SINGLE_MACHINE:** Change to **N**.
- iii. **SSL_ENABLED:** Change to **true** to enable SSL. (Two-way SSL for Elasticsearch. One-way SSL for Kafka and Tomcat services)
- iv. **SSL_TWO_WAY_ENABLED:** Change to **true**, if you want to enable two-way SSL for Tomcat services.

Example:

```
### Installation Details ###
USER_INSTALL_DIR=/home/<username>/Analytics
SINGLE_MACHINE=N
SSL_ENABLED=true
SSL_TWO_WAY_ENABLED:true
```

b. Set the following SSL-related values:



Ensure that the user who is installing CA Jarvis has read permission to the location where the Keystore and Truststore files reside.

KEYSTORE_FILEPATH: Path where the keystore.jks file is located.

TRUSTSTORE_FILEPATH: Path where the truststore.jks file is located.

KEYSTORE_PASSWORD: The password that is used to access the keystore file when SSL is enabled. Use the same password that you used while you have created the certificates.

TRUSTSTORE_PASSWORD: The password that is used to access the truststore file when SSL is enabled. Use the same password that you used while you have created the certificates.

KEY_PASSWORD: The password that is used to access the key when SSL is enabled. Use the same password that you used while you have created the certificates.

Example:

```
KEYSTORE_FILEPATH=/home/qa/jarvis/sslCerts/CN=ciavrovrm-keystore.jks
```

```
TRUSTSTORE_FILEPATH=/home/qa/jarvis/sslCerts/truststore.jks
KEYSTORE_PASSWORD=1237c3b0ef108fe28b90
TRUSTSTORE_PASSWORD=dc075fba9dd181b77fb1
KEY_PASSWORD=abcd1234
```

- c. Uncomment (remove #) the following line and provide absolute path to the mounted data directory, if you want to change the default location. Ensure that <username> has read/write permissions to the mounted data directory as mentioned in step 9.

```
#KAFKA_MOUNT=$USER_INSTALL_DIR$/Data/kafkaMNT
#ZOOKEEPER_MOUNT=$USER_INSTALL_DIR$/Data/zkMNT
Example:
#KAFKA_MOUNT=/home/<username>/Analytics/Data/kafkaMNT
#ZOOKEEPER_MOUNT=/home/<username>/Analytics/Data/zkMNT
```

- d. Provide the Kafka, Zookeeper, and Elasticsearch installation details.



Separate the IP addresses or host-names by using comma(,).

When you set up properties like KAFKA_INSTALLATION_HOST_PORTS, and so on, provide the same IP addresses or Fully-Qualified Domain Names (FQDNs) that you have used while you have set up SSL certificates.

Provide only FQDN (not IP address) for installing Jarvis on multiple nodes with SSL enabled. FQDN is case-sensitive.

- i. **KAFKA_INSTALLATION_HOST_PORTS:** Provide the IP address or FQDN and port number (9092) of the host where Kafka should be installed. If you do not provide this value, Kafka will not be installed.

Use the following command to find the IP address. You can find the IP address after "inet adr" in the output.

```
/sbin/ifconfig
```

Use the following command if the previous command fails because of privilege-related error:

```
sudo /sbin/ifconfig
```

- ii. **ZOOKEEPER_INSTALLATION_HOST_PORTS:** Provide the IP address or FQDN and port number (2181) of the host where ZOOKEEPER should be installed.

- iii. **SCHEMA_REGISTRY_INSTALLATION_HOSTS=** Provide the IP address or FQDN and port number (8081) of the host where schema registry should be installed. If you do not provide this value, schema registry will not be installed. There should not be spaces between host names.

- iv. **ES_HOST_MASTER_NODE:** Provide the IP address or FQDN and port number of the ElasticSearch master node. Use **9300** as the port number. (TCP transport port)

- v. **ES_HOST_DATA_NODE**: Provide the IP address or FQDN and port number of the ElasticSearch data node. Use **9300** as the port number. (TCP transport port)
- vi. **ES_HOST_CLIENT_NODE**: Provide the IP address or FQDN and port number of the ElasticSearch client node. Use **9300** as the port number. (TCP transport port)
- vii. **ES_HOST_HOT_NODE**: Provide the IP address or FQDN and port number of the ElasticSearch hot node. Use **9300** as the port number. (TCP transport port)



It is mandatory to have at least one hot node configured.

- viii. **ES_HOST_WARM_NODE**: Provide the IP address or FQDN and port number of the ElasticSearch warm node. Use **9300** as the port number.

e. Provide the following details:

- i. **ES_CLUSTER_NAME**: Provide a name for the ElasticSearch cluster.
- ii. **ES_DATA_MOUNT**: Provide absolute path to the mounted data directory. Ensure that <username> has read/write permissions to the mounted data directory as mentioned in step 9.



During an upgrade, the data mount directory may get removed. Ensure that you provide a path outside the Elasticsearch directory as the value for ES_DATA_MOUNT.

Example:

```
#ES_DATA_MOUNT=/home/<username>/Analytics/Data/es
```

- iii. **TOMCAT_INSTALLATION_HOST**=host1:port1: Provide the IP address or FQDN and port number (8443) of the host where Tomcat should be installed.
- iv. **VERIFIER_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Verifier should be installed.
- v. **INDEXER_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Indexer should be installed.
- vi. **ESUTILS_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Elasticsearch utilities should be installed.
- vii. **KRON_HOST**=localhost:8443: Provide the IP address or FQDN of the host where KRON should be installed.

- f. (Optional) Update the following variables to define the total number of Kafka consumers in Verifier and Indexer. PRIORITY_RATIO is calculated based on these values. For example, if the count is defined as 100, value of P1 will be 80, value of P2 will be 10, and value of P3 will be 10.

```
### Total no of kafka consumer in Verifier
VERIFIER_KAFKA_CONSUMER_COUNT=4
### Total no of kafka consumer in Indexer
INDEXER_KAFKA_CONSUMER_COUNT=4
```

- g. (Optional) Uncomment the following line to define the priorities for data ingestion.

```
#Priority ratio
#PRIORITY_RATIO=P1:80,P2:10,P3:10
```



Example:

PRIORITY_RATIO= P1:70,P2:20,P3:10

In this example, **P1**, **P2**, and **P3** are the priorities for the messages to be processed. You can define up to nine levels of priorities. 70,20, and 10 are the percentage of resources that must be allocated to each priority bucket (topics in kafka) for processing. The sum of the percentages should not exceed 100.

If a machine or container has 4 cores, then there are 8 threads /consumers that will be divided proportionately according to the priority ratio.

PRIORITY_RATIO= P1:70,P2:20,P3:10 and number of cores = 4.

There are 8 threads/consumers in the resource pool. So, approximately P1 gets 6 consumers , P2 gets 1 consumer, and P3 gets 1 consumer.

While sending the data (ingestion), you can provide the priority (**priority**) in the HTTP header of the input messages. Example: curl -XPOST -H '**priority: p3**' <ingestion url>

If **PRIORITY_RATIO** is not set, then by default all the data will be handled by P1 priority and all the resources will be allocated to it.

- h. (Optional) Update the following details:

```
#ES SNAPSHOT location
#ES_SNAPSHOT_LOCATION=/opt/CA/Analytics/Data/backup : The location
where Elasticsearch stores snapshot.
#Exclude indices from snapshot
#ES_SNAPSHOT_EXCLUDE_INDICES=index_list : The list of indexes that
should be excluded during back up.
#Datapurge retention period.
#DATAPURGE_DEFAULT_RETENTION_PERIOD=45 : The number of days up to
which the data should be kept.
#ES_SNAPSHOT_REPOSITORY_NAME=RepositoryName: Before any snapshot or
restore operation can be performed, a snapshot repository should be
registered in Elasticsearch. This variable indicates the name of such
a repository.
#ES_SNAPSHOT_REPOSITORY_NAME=RepositoryName: Before any snapshot or
restore operation can be performed, a snapshot repository should be
registered in Elasticsearch. This variable indicates the name of such
```

a repository.

#INDEX_LIMIT:NUMBER_OF_SHARDS_PER_NODE: Assign the number of shards that will be present on each node of the Elasticsearch cluster. By restricting the number of shards allocated to each node, you can distribute the load across the Elasticsearch cluster.

This value is directly dependent on the following values:

- Number of available nodes.
- Number of primary shards present for the index.

If the value is very low, it affects the allocation of the primary shards. If the value is very high, it degrades the performance of the node.

If you do not assign any value for this parameter, the system calculates the value based on the number of available nodes and primary shards. Even if you have assigned a value for this parameter, the system checks the assigned value against the value calculated by the system. It then uses the value that suits better for your cluster.

Example:

Your setup is as follows:

- Number of available nodes = 3
- Number of primary shards present for the index = 5 (Replication factor=1)

The system determines that the suitable shards per node is 5. By doing so, the cluster will be safe even in the case of a node failure. In this example, using a very low value like 1 (primary shards themselves will be unassigned) or a very high value like 10 (results in an imbalance within the cluster) is not recommended.

i. Provide the following details if your CA Jarvis Dashboard setup is SSL-enabled:

i. **LD DS_SSL_CONFIGURED=false**. Change to **true** if CA Jarvis Dashboard is SSL-enabled.

ii. **LD DS_HOST=localhost:8080**. Provide the IP address or FQDN of the CA Jarvis Dashboard host.

15. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes made to the `analyticsInstaller.properties` file.

16. Change to the parent directory :

Example:

```
cd /home/<username>/jarvisInstaller_installAnyWhere
```

17. Run the following commands to install Jarvis:

```
./CA_Analytics.bin -f <path>/analyticsInstaller.properties
```

Example:

```
./CA_Analytics.bin -f properties/analyticsInstaller.properties
```

The installer displays the progress of installation. For multiple nodes, you have to [manually run the Jarvis Health-check](#).

Jarvis is installed as per the configurations in the **analyticsInstaller.properties** file. The On successful installation, the installer starts all the installed services.



If you get a message saying that port is already in use, change the port number in the `analyticsInstaller.properties` file and try to install again.

18. **(Optional)** Go to the following directory to see the log files generated by the installer:

<USER_INSTALL_DIR>/_CA Analytics_installation/Logs

19. **(Optional:)** Execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Command Format for starting and stopping services:

```
sudo systemctl <command><serviceName>
```

Command Format for checking the status of services:

```
systemctl <command> <serviceName>
```

Where <command> can be one of the following:

```
start
stop
status
```

And, <serviceName> can be one of the following:

ca_zookeeper.service should be started before starting **ca_kafka.service**.

After starting **ca_kafka.service**, the rest of the services can be started in any order.

```
ca_zookeeper.service
ca_kafka.service
ca_schema-registry.service
ca_elasticsearch.service
ca_tomcat.service
ca_verifier-ingest.service
ca_indexer-ingest.service
ca_esutils.service
```

Applies for non-root non-sudo user) Change to <USER_INSTALL_DIR>/bin and execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Applies for non-root non-sudo user) Change to <USER_INSTALL_DIR>/bin and execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Command for starting the Jarvis services:

```
./startElasticsearch.sh
./startZookeeper.sh
./startkafka.sh
./startSchemaRegistry.sh
./startIndexerIngest.sh
./startIndexerUpdate.sh
./startVerifierIngest.sh
./startVerifierUpdate.sh
./startESUtils.sh
./startTomcat.sh
```

Command for stopping the Jarvis services:

```
./stopElasticsearch.sh
./stopSchemaRegistry.sh
./stopZookeeper.sh
./stopkafka.sh
./stopIndexerIngest.sh
./stopIndexerUpdate.sh
./stopVerifierIngest.sh
./stopVerifierUpdate.sh
./stopESUtils.sh
./stopTomcat.sh
```

Installed Jarvis. What Next?

You are now ready to start [using Jarvis APIs](#).

CA Jarvis - Non SSL

The following topic explains the steps to install CA Jarvis on multiple nodes. You can use the same procedure to install Jarvis on the Amazon Web Services (AWS) platform.



A root user must download the installation files and create a non-root sudo user. As a non-root sudo user, you can continue the installation once the root user performs his steps.

We recommend you to install Jarvis as a **non-root sudo user** so that Jarvis will be installed as a service. You can use the systemctl commands to start and stop the services.

If you are installing Jarvis as a non-root non-sudo user, Jarvis will be installed as an application. You must use shell scripts to start and stop the components. You must follow the random steps marked for non-root non-sudo user to complete the installation as a non-root non-sudo user.

Prerequisites

Software and Hardware Requirements



The Minimum values mentioned here are for development and test environments, and the Recommended values are for production environment.

Component	Platform	Memory		Processor		Hard Disk	
		Minimum	Recommended	Minimum	Recommended	Minimum	Recommended
Kafka & Zookeeper	▪ RHEL 7.2	8 GB	32 GB	2-Core	4-Core	100 GB Solid State Device (SSD)	▪ 800 GB SSD
	▪ CentOS 7.2 or above.						

Component	Platform	Memory		Processor		Hard Disk	
Elasticsearch	<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above. 	8 GB	64 GB	4-Core	8-Core	100 GB SSD	800 GB SSD with RAID0 format
Verifier	<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above. 	4 GB	8 GB	2-Core	4-Core	100 GB SSD	100 GB SSD
Indexer	<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above. 	4 GB	8 GB	2-Core	4-Core	100 GB SSD	100 GB SSD
Apache Tomcat	<ul style="list-style-type: none"> ▪ RHEL 7.2 	4 GB	8 GB	2-Core	4-Core	100 GB SSD	100 GB SSD

Component	Platform	Memory	Processor	Hard Disk
-----------	----------	--------	-----------	-----------

- CentOS 7.2 or above.

Other Requirements

- Ensure that the **/etc/hosts** in all your servers have the correct value set in the following format:

<IP Address> <Host Name with domain> <Alias>

Example:

10.131.58.124 user05-I193354.ca.com user05-I193354

For non-root and non-sudo Users

- Ensure that *ulimit* is set to 65536 and *vm.max_map_count* is set to 262144 for Elasticsearch to start successfully by completing the following procedures:

Do the following steps as a root user to set ulimit

1.
 - a.
 - i. Change to the */etc/security* directory.
 - ii. Add the following lines to the *limits.conf* file:

```
* ulimit -n 65536
* hard nproc 65536
* soft nproc 65536
* soft nofile 65536
* hard nofile 131072
```

Do the following steps as a root user to set vm.max_map_count:

1.
 - a. Change to the */etc* directory.
 - b. Update the *sysctl.conf* file with the following value:

```
vm.max_map_count=262144
```
 - c. Run the following command to get the changes reflected without restarting your node:

```
sysctl -q -w vm.max_map_count=262144
```

- Ensure that your firewall does not block CA Jarvis from the ports it requires. Ensure that the following ports are open on all the machines where you are installing CA Jarvis:

- Apache Tomcat: **8080** (SSL Disabled) and **8443** (SSL Enabled)
- Apache Kafka: **9092** and **9093**
- Apache Zookeeper: **2181**, **2888**, and **3888**
- Elasticsearch: **9300**

Run the following commands to list the open ports:

You can skip the first command if Network Mapper (nmap) is already installed.

```
yum install nmap
nmap localhost
```



Ensure that your keyboard and display locale are set to English (UTF-8) only. As a root user, type the following command to see the current locale:

```
locale
```

Installation Instructions

The following procedure explains the steps to install Jarvis on multiple nodes.



You can find the errors that might happen during the installation, and the possible resolutions [here](#).

Refer the [Troubleshooting](#), if you encounter issues during the installation.

In this procedure, the term "ports" indicates Transport Control Protocol (TCP) ports.

1. Log in as a root user.
2. Download latest version of the Jarvis installer package (.tar file) from CA Support site.
3. Copy the .tar file to the target machine. For example, copy the file to the `/tmp` folder in the target machine.
4. Run the following command to untar the file. Ensure that you use the correct name for the .tar file.

```
cd /tmp; tar -xvf jarvisInstaller_installAnywhere-<version>.tar
```

This command creates `/<location of the tar file>/jarvisInstaller_installAnywhere`, which contains the installation files and Jarvis binaries.

5. Ignore step 5 and 6, if you have already created a non-root user, and want to use the same user for installing Jarvis. If want to create a non-root user for installing Jarvis, run the following command and continue with step 6:

```
cd jarvisInstaller_installAnywhere/scripts
```

6. Create a **non-root sudo user** by running the following commands. From step 10, you will be using the non-root-user credentials that you have set in this step.

```
chmod +x prepareMachineAsRoot.sh
./prepareMachineAsRoot.sh <username>
```

<username> must be a string.

For non-root and non-sudo Users

Create a **non-root non-sudo user** by running the following command. Ensure that you customize the command and provide password for the non-root user when prompted:

```
useradd <username>
```

<username> must be a string.

7. Run the following command:

```
cd /tmp
```

8. Run the following commands one by one:

```
chown -R <username>:<username> jarvisInstaller_installAnywhere
```

```
chmod +x jarvisInstaller_installAnywhere/CA_Analytics.bin
```

9. Ensure that <username> has the read/write permissions to the directories that are going to be used during the installation. This is required if you are using a directory outside of <username>'s home directory to set values for USER_INSTALL_DIR, KAFKA_MOUNT, or ES_DATA_MOUNT. To set permissions, complete the following steps:

a. Change to the directory to which you want to set the permissions.

b. Customize and run the following commands:

```
chmod 755 <name of the directory>
```

```
chown -R <username>:<username> <name of the directory>
```

10. Log in as the non-root user to whom the permissions were set in step 6.

```
su - <username>
```

11. Copy the jarvisInstaller_installAnywhere directory to the home directory of <username> (/home/<username>):

Example:

```
cp -r /tmp/jarvisInstaller_installAnyWhere /home/<username>/.
```

12. Run the following command:

```
cd jarvisInstaller_installAnyWhere/properties
```

13. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi analyticsInstaller.properties
```

14. Make the following changes in the **analyticsInstaller.properties** file:



Because this file contains passwords, ensure that you have elevated permissions for this file.

If you are editing a commented value, for example, #INSTALL_APIIS=true, ensure that you uncomment (remove #) while editing.

Ensure that there are no additional spaces after the values entered in the **analyticsInstaller.properties** file. For example, there should not be any spaces after the Value Y when you set the following parameter:

```
SINGLE_MACHINE=Y.
```

- a. Provide the following information:

- i. **USER_INSTALL_DIR:** Provide the location where Jarvis should be installed. Ensure that <username> has read/write permissions to this directory as mentioned in step 9.
- ii. **SINGLE_MACHINE:** Change to **N**.

Example:

```
### Installation Details ###
USER_INSTALL_DIR=/home/<username>/Analytics
SINGLE_MACHINE=N
SSL_ENABLED=false
```

- b. Uncomment (remove #) the following line and provide absolute path to the mounted data directory, if you want to change the default location. Ensure that <username> has read/write permissions to the mounted data directory as mentioned in step 9.

```
#KAFKA_MOUNT=$USER_INSTALL_DIR$/Data/kafkaMNT
#ZOOKEEPER_MOUNT=$USER_INSTALL_DIR$/Data/zkMNT
Example:
#KAFKA_MOUNT=/home/<username>/Analytics/Data/kafkaMNT
#ZOOKEEPER_MOUNT=/home/<username>/Analytics/Data/zkMNT
```

- c. Provide the Kafka, Zookeeper, and Elasticsearch installation details.



Separate the IP addresses or host-names by using comma (,).

When you set up properties like `KAFKA_INSTALLATION_HOST_PORTS`, and so on, provide the same IP addresses or Fully-Qualified Domain Names (FQDNs) that you have used while you have set up SSL certificates.

Provide either FQDN or IP address for installing Jarvis on multiple nodes without SSL. FQDN is case-sensitive.

- i. **KAFKA_INSTALLATION_HOST_PORTS**: Provide the IP address or FQDN and port number (9092) of the host where Kafka should be installed. If you do not provide this value, Kafka will not be installed.

Use the following command to find the IP address. You can find the IP address after "inet adr" in the output.

```
/sbin/ifconfig
```

Use the following command if the previous command fails because of privilege-related error:

```
sudo /sbin/ifconfig
```

- ii. **ZOOKEEPER_INSTALLATION_HOST_PORTS**: Provide the IP address or FQDN and port number (2181) of the host where ZOOKEEPER should be installed.
- iii. **SCHEMA_REGISTRY_INSTALLATION_HOSTS**: Provide the IP address or FQDN and port number (8081) of the host where schema registry should be installed. If you do not provide this value, schema registry will not be installed. There should not be spaces between host names.
- iv. **ES_HOST_MASTER_NODE**: Provide the IP address or FQDN and port number of the ElasticSearch master node. Use **9300** as the port number (TCP transport port).
- v. **ES_HOST_DATA_NODE**: Provide the IP address or FQDN and port number of the ElasticSearch data node. Use **9300** as the port number. (TCP transport port)
- vi. **ES_HOST_CLIENT_NODE**: Provide the IP address or FQDN and port number of the ElasticSearch client node. Use **9300** as the port number. (TCP transport port)
- vii. **ES_HOST_HOT_NODE**: Provide the IP address or FQDN and port number of the ElasticSearch hot node. Use **9300** as the port number (TCP transport port).



It is mandatory to have at least one hot node configured.

- viii. **ES_HOST_WARM_NODE**: Provide the IP address or FQDN and port number of the ElasticSearch warm node. Use **9300** as the port number. (TCP transport port)

- d. Provide the following details:

- i. **ES_CLUSTER_NAME**: Provide a name for the ElasticSearch cluster.
- ii. **ES_DATA_MOUNT**: Provide an absolute path to the mounted data directory. Ensure that <username> has read/write permissions to the mounted data directory as mentioned in step 9.



During an upgrade, the data mount directory may get removed. Ensure that you provide a path outside the Elasticsearch directory as the value for ES_DATA_MOUNT.

Example:

```
#ES_DATA_MOUNT=/home/<username>/Analytics/Data/es
```

- iii. **TOMCAT_INSTALLATION_HOST**=host1:port1: Provide the IP address or FQDN and port number (8080) of the host where Tomcat should be installed.
 - iv. **VERIFIER_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Verifier should be installed.
 - v. **INDEXER_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Indexer should be installed.
 - vi. **ESUTILS_INSTALLATION_HOSTS**=host1,host2: Provide the IP address or FQDN of the host where Elasticsearch utilities should be installed.
 - vii. **KRON_HOST**=localhost:8080: Provide the IP address or FQDN of the host where KRON should be installed.
- e. (Optional) Update the following variables to define the total number of Kafka consumers in Verifier and Indexer. PRIORITY_RATIO is calculated based on these values. For example, if the count is defined as 100, value of P1 will be 80, value of P2 will be 10, and value of P3 will be 10.

```
### Total no of kafka consumer in Verifier
VERIFIER_KAFKA_CONSUMER_COUNT=4
### Total no of kafka consumer in Indexer
INDEXER_KAFKA_CONSUMER_COUNT=4
```

- f. (Optional) Uncomment the following line to define the priorities for data ingestion.

```
#Priority ratio
#PRIORITY_RATIO=P1:80,P2:10,P3:10
```



Example:

PRIORITY_RATIO= P1:70,P2:20,P3:10

In this example, **P1**, **P2**, and **P3** are the priorities for the messages to be processed. You can define up to nine levels of priorities. 70,20, and 10 are the percentage of resources that must be allocated to each priority bucket (topics in kafka) for processing. The sum of the percentages should not exceed 100.

If a machine or container has 4 cores, then there are 8 threads /consumers that will be divided proportionately according to the priority ratio.

PRIORITY_RATIO= P1:70,P2:20,P3:10 and number of cores = 4.

There are 8 threads/consumers in the resource pool. So, approximately P1 gets 6 consumers , P2 gets 1 consumer, and P3 gets 1 consumer.

While sending the data (ingestion), you can provide the priority (**priority**) in the HTTP header of the input messages. Example: curl -XPOST -H '**priority: p3**' <ingestion url>

If **PRIORITY_RATIO** is not set, then by default all the data will be handled by P1 priority and all the resources will be allocated to it.

g. (Optional) Update the following details:

```
#ES_SNAPSHOT location
#ES_SNAPSHOT_LOCATION=/opt/CA/Analytics/Data/backup : The location
where Elasticsearch stores snapshot.
#Exclude indices from snapshot
#ES_SNAPSHOT_EXCLUDE_INDICES=index_list : The list of indexes that
should be excluded during back up.
#Datapurge retention period.
#DATAPURGE_DEFAULT_RETENTION_PERIOD=45 : The number of days up to
which the data should be kept.
#ES_SNAPSHOT_REPOSITORY_NAME=RepositoryName: Before any snapshot or
restore operation can be performed, a snapshot repository should be
registered in Elasticsearch. This variable indicates the name of such
a repository.
#ES_SNAPSHOT_REPOSITORY_NAME=RepositoryName: Before any snapshot or
restore operation can be performed, a snapshot repository should be
registered in Elasticsearch. This variable indicates the name of such
a repository.
#INDEX_LIMIT:NUMBER_OF_SHARDS_PER_NODE: Assign the number of shards
that will be present on each node of the Elasticsearch cluster.
By restricting the number of shards allocated to each node, you can
distribute the load across the Elasticsearch cluster.
This value is directly dependent on the following values:
• Number of available nodes.
• Number of primary shards present for the index.
If the value is very low, it affects the allocation of the primary
shards. If the value is very high, it degrades the performance of the
node.
If you do not assign any value for this parameter, the system
calculates the value based on the number of available nodes and
primary shards. Even if you have assigned a value for this parameter,
the system checks the assigned value against the value calculated by
the system. It then uses the value that suits better for your cluster.
Example:
Your setup is as follows:
• Number of available nodes = 3
• Number of primary shards present for the index = 5 (Replication
factor=1)
The system determines that the suitable shards per node is 5. By
doing so, the cluster will be safe even in the case of a node failure.
In this example, using a very low value like 1 (primary shards
themselves will be unassigned) or a very high value like 10 (results
in an imbalance within the cluster) is not recommended.
```

15. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes made to the analyticsInstaller.properties file.

16. Change to the parent directory :

Example:

```
cd /home/<username>/jarvisInstaller_installAnywhere
```

17. Run the following commands to install Jarvis:

```
./CA_Analytics.bin -f <path>/analyticsInstaller.properties
```

Example:

```
./CA_Analytics.bin -f properties/analyticsInstaller.properties
```

The installer displays the progress of installation. For the installation on multiple nodes, you have to [manually run the Jarvis Health-check](#).

Jarvis is installed as per the configurations in the **analyticsInstaller.properties** file. The On successful installation, the installer starts all the installed services.



If you get a message saying that port is already in use, change the port number in the analyticsInstaller.properties file and try to install again.

18. **(Optional)** Go to the following directory to see the log files generated by the installer:

```
<USER_INSTALL_DIR>/_CA Analytics_installation/Logs
```

19. **(Optional:)** Execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Command Format for starting and stopping services:

```
sudo systemctl <command><serviceName>
```

Command Format for checking the status of services:

```
systemctl <command> <serviceName>
```

Where **<command>** can be one of the following:

```
start
stop
status
```

And, **<serviceName>** can be one of the following:

ca_zookeeper.service should be started before starting **ca_kafka.service**. After starting **ca_kafka.service**, the rest of the services can be started in any order.

```
ca_zookeeper.service
ca_kafka.service
ca_schema-registry.service
ca_elasticsearch.service
ca_tomcat.service
ca_verifier-ingest.service
ca_indexer-ingest.service
ca_esutils.service
```

Applies for non-root non-sudo user) Change to **<USER_INSTALL_DIR>/bin** and execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the installation.

Applies for non-root non-sudo user) Change to **<USER_INSTALL_DIR>/bin** and execute the following commands to start or stop the Jarvis services. Remember that the installer starts all the Jarvis services after the

installation.**Command for starting the Jarvis services:**

```
./startElasticsearch.sh
./startZookeeper.sh
./startkafka.sh
./startSchemaRegistry.sh
./startIndexerIngest.sh
./startIndexerUpdate.sh
./startVerifierIngest.sh
./startVerifierUpdate.sh
./startESUtils.sh
./startTomcat.sh
```

Command for stopping the Jarvis services:

```
./stopElasticsearch.sh
./stopSchemaRegistry.sh
./stopZookeeper.sh
./stopkafka.sh
./stopIndexerIngest.sh
./stopIndexerUpdate.sh
./stopVerifierIngest.sh
./stopVerifierUpdate.sh
./stopESUtils.sh
./stopTomcat.sh
```

Installed Jarvis. What Next?

You are now ready to start [using Jarvis APIs](#).

Installation of CA Jarvis Dashboard

- [Installation of CA Jarvis Dashboard on a Single Node \(Without High-Availability Failover\)](#)
- [Installation of CA Jarvis Dashboard on a Single Node \(With High-Availability Failover\)](#)
- [Installation of CA Jarvis Dashboard on a Single Node: Couchdb \(Without High-Availability Failover\)](#)

Installation of CA Jarvis Dashboard on a Single Node (Without High-Availability Failover)

- [Why Should I Install CA Jarvis Dashboard?](#)
- [Prerequisites](#)
 - [Software and Hardware Requirements](#)
 - [\(Optional\) SSL Requirements](#)
 - [Other Requirements](#)
- [Installation Instructions](#)
- [Installed CA Jarvis Dashboard What Next?](#)

Why Should I Install CA Jarvis Dashboard?

CA Jarvis Dashboard helps you visualize real-time analytics by creating comprehensive business reports. CA Jarvis Dashboard is a visualization platform that is designed to search, view, and interact with the data that is stored in CA Jarvis. So, to start visualizing the data that is stored in CA Jarvis, you should install CA Jarvis Dashboard.

Prerequisites

Software and Hardware Requirements

Platform	Memory	Processor	Hard Disk	Others
<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above 	4 GB	2-Core	Hard disk: Minimum of 100-GB space	Unzip utility.

(Optional) SSL Requirements

To enable SSL, ensure that you have the following values after setting up the SSL certificates:

- Location where keystore.jks file and truststore.jks file are stored
- The keystore and truststore passwords
- Fully Qualified Domain Name (FQDN)

Other Requirements

- Ensure that the **/etc/hosts** in all your servers have the correct value set in the following format:
<IP Address> <Host Name with domain> <Alias>
Example:10.131.58.124 user05-l193354.ca.com user05-l193354
- Ensure that your firewall does not block CA Jarvis Dashboard from the ports it requires. Ensure that the following ports are open on the node where you are installing CA Jarvis Dashboard:
Apache Tomcat: **8080** (SSL Disabled) and **8443** (SSL Enabled.)
- Collect the IP address and Subnetmask address for the node.

Installation Instructions

The following procedure explains the steps to install CA Jarvis Dashboard on a single node.

1. Log in as a root user.
2. Download latest version of the CA Jarvis Dashboard installer package (.zip file) from CA Support site.
3. Copy the .zip file to the target machine. For example, copy the file to the */tmp* folder in the target machine.
4. Run the following command to unzip the file. Ensure that you use the correct name for the .zip file.

```
cd /tmp; unzip CALDDS-<version>.zip
```

This command creates */<location of the zip file>/CALDDS-Installer*, which contains the installation files.

5. **(This step is required if you are using a directory outside of <username>'s home directory to set values for *USER_INSTALL_DIR*.)** Ensure that <username> has the read/write permissions to the directories that are going to be used during the installation.

6. Run the following command:

```
cd CALDDS-Installer
```

7. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi lddsinstaller.properties
```

8. Make the following changes in the **lddsinstaller.properties** file:



Because this file contains passwords, ensure that you have elevated permissions for this file.

If you are editing a commented value, for example, #USER_INSTALL_DIR, ensure that you uncomment (remove #) while editing.

No additional spaces are allowed after the values entered in the **lddsinstaller.properties** file. For example, there should not be any spaces after the Value Y when you set the following parameter:

```
SINGLE_MACHINE=Y
```

- a. Provide the following information:

i. **SINGLE_MACHINE**: Keep Y.

ii. **USER_INSTALL_DIR**: Provide the location where CA Jarvis Dashboard should be installed.

- b. Provide the information regarding database configuration:

i. **DATABASE_PORT**: Provide the PostgreSQL port number for your node.

ii. **DATABASE_DEFAULT_USER_PASSWORD**: Provide the password for PostgreSQL (the default user of PostgreSQL .)

iii. **DATABASE_NEW_USER_USERNAME**: Provide the user name for the owner of CA Jarvis Dashboard database.

iv. **DATABASE_NEW_USER_PASSWORD**: Provide the password for the owner of CA Jarvis Dashboard database.

v. **LD DS_DATABASE_NAME**: Provide a name for the CA Jarvis Dashboard database.

- c. Provide the information regarding the Operating system user:

i. **OS_USER_USERNAME**: Provide user name for a non-root user. PostgreSQL does not allow a root user to do the configurations. To configure PostgreSQL database, we are creating non root user and switching to that user during the installation.

- d. Provide the Email configuration details:

- i. **EMAIL_HOST**: SMTP Host where the email server is configured. Provide valid email host to get email when the forgot password option is selected.
 - ii. **EMAIL_PORT**: Port on the host where the email server is configured.
 - iii. **EMAIL_USERNAME**: Provide the email address for the user for whom the email is configured.
 - iv. **EMAIL_PASSWORD**: Provide password to access the email account.
 - v. **EMAIL_FROMNAME**: Provide a label to identify the sender of the email.
 - vi. **EMAIL_HOSTNAME**:
- e. Provide the SSL-related information:
- i. **SSL_ENABLED** : Change to **true**, if you want to enable SSL.
 - ii. **SSL_TWO_WAY_ENABLED**: Keep **false**, if you want to enable one-way SSL. Change to **true**, if you want to enable two-way SSL.
 - iii. **SSL_PORT**: Provide the HTTPS port number.
 - iv. **KEYSTORE_FILEPATH**: Path where the keystore.jks file is located.
 - v. **TRUSTSTORE_FILEPATH**: Path where the truststore.jks file is located.
 - vi. **KEYSTORE_PASSWORD**: The password that is used to access the keystore file when SSL is enabled. Use the same password that you used while you have created the certificates.
 - vii. **TRUSTSTORE_PASSWORD**: The password that is used to access the truststore file when SSL is enabled. Use the same password that you used while you have created the certificates.
9. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes that are made to the lddsinstaller.properties file.
10. Run the following commands to install CA Jarvis Dashboard:

```
./setup.bin -f <path>/lddsinstaller.properties
```

Example: `./setup.bin -f lddsinstaller.properties`

The installer displays the progress of installation.

CA Jarvis Dashboard is installed as per the configurations in the lddsinstaller.properties file. The On successful installation, the installer starts all the installed services.



If you get a message saying that port is already in use, change the port number in the lddsinstaller.properties file and try to install again.

11. **(Optional)** Go to the following directory to see the log files that are generated by the installer:

`<USER_INSTALL_DIR>/logs`



If the installation fails, refer the `/tmp` directory for logs.

12. **(Optional:)** Change to `<USER_INSTALL_DIR>/bin` and execute the following commands to start or stop the CA Jarvis Dashboard services. Remember that the installer starts all the CA Jarvis Dashboard services after the installation.

To start the tomcat and postgresSQL database:

`./startservers.sh`

To stop the tomcat and postgresSQL database:

`./stopservers.sh`

Installed CA Jarvis Dashboard What Next?

Refer the following article to onboard product and tenant.

[API Reference of CA Jarvis Dashboard.](#)

After onboarding product and tenant, you are ready to use CA Jarvis Dashboard.

Installation of CA Jarvis Dashboard on a Single Node (With High-Availability Failover)

- [Why Should I Install CA Jarvis Dashboard?](#)
- [Why Should I Enable High-Availability Failover?](#)
- [Prerequisites](#)
 - [Software and Hardware Requirements](#)
 - [\(Optional\) SSL Requirements](#)
 - [Other Requirements](#)
- [Installation Instructions](#)
- [Installed CA Jarvis Dashboard. What Next?](#)

Why Should I Install CA Jarvis Dashboard?

The CA Jarvis Dashboard helps you visualize real-time analytics by creating comprehensive business reports. The CA Jarvis Dashboard is a visualization platform that is designed to search, view, and interact with the data that is stored in CA Jarvis. So, to start visualizing the data that is stored in CA Jarvis, you should install the CA Jarvis Dashboard.

Why Should I Enable High-Availability Failover?

If the primary system fails, high-availability failover helps switch automatically to a redundant backup system. Switching to a backup system ensures reliable and continuous service of the CA Jarvis Dashboard on your environment.

Prerequisites

Software and Hardware Requirements

Two nodes of the following configuration are required.

Platform	Memory	Processor	Hard Disk	Others
<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above 	4 GB	2-Core	Hard disk: Minimum of 100-GB space	Unzip utility.

(Optional) SSL Requirements

To enable SSL, ensure that you have the following values after setting up the SSL certificates:

- Location where keystore.jks file and truststore.jks file are stored
- The keystore and truststore passwords
- Fully Qualified Domain Name (FQDN)

Other Requirements

- **(For both the nodes)** Ensure that the `/etc/hosts` in all your servers have the correct value set in the following format:
`<IP Address> <Host Name with domain> <Alias>`
Example: 10.131.58.124 [user05-l193354.ca.com](#) user05-l193354
- **(Only for the first node)** Ensure that your firewall does not block CA Jarvis Dashboard from the ports it requires. Ensure that the following ports are open on all the machines where you are installing the CA Jarvis Dashboard:
 Apache Tomcat: **8080** (SSL Disabled) and **8443** (SSL Enabled.)
- Collect the IP address and Subnetmask address for both the nodes.
- Ensure that both nodes are mutually accessible.

Installation Instructions

The following procedure explains the steps to install the CA Jarvis Dashboard on a single node with high-availability failover enabled. This procedure requires two nodes (Node1 and Node2) of the same configuration.

1. Log in as a root user on Node1.
2. Download latest version of the CA Jarvis Dashboard installer package (.zip file) from [CA Support](#) site.
3. Copy the .zip file to the target machine. For example, copy the file to the `/tmp` folder in the target machine.
4. Run the following command to unzip the file. Ensure that you use the correct name for the .zip file.

```
cd /tmp; unzip CALDDS-<version>.zip
```

This command creates */<location of the zip file>/CALDDS-Installer*, which contains the installation files.

5. **(This step is required if you are using a directory outside of <username>'s home directory to set values for *USER_INSTALL_DIR*.)** Ensure that <username> has the read /write permissions to the directories that are going to be used during the installation.

6. Run the following command:

```
cd CALDDS-Installer
```

7. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi lddsinstaller.properties
```

8. Make the following changes in the **lddsinstaller.properties** file:



Because this file contains passwords, ensure that you have elevated permissions for this file.

If you are editing a commented value, for example, `#USER_INSTALL_DIR`, ensure that you uncomment (remove #) while editing.

No additional spaces are allowed after the values entered in the **lddsinstaller.properties** file. For example, there should not be any spaces after the Value Y when you set the following parameter:

```
SINGLE_MACHINE=N
```

- a. Provide the following information:

- i. **SINGLE_MACHINE:** Change to **N**

- ii. **USER_INSTALL_DIR:** Provide the location where CA JarvisDashboard should be installed.

- b. Provide the information regarding high-availability configuration:

- i. **LDDS_HANODES_HOST:** Provide the IP address of both Node1 and Node2 separated by comma.

- ii. **LDDS_HANODES_SUBNET_MASK:** Provide the Subnet mask address of both Node1 and Node2 separated by comma.

- iii. **DATABASE_PORT:** Provide the PostgreSQL port number for Node1 and Node2 separated by comma.

- iv. **DATABASE_DEFAULT_USER_PASSWORD:** Provide the password for PostgreSQL (the default user of PostgreSQL .)

- v. **DATABASE_NEW_USER_USERNAME**: Provide the user name for the owner of the CA Jarvis Dashboard database.
 - vi. **DATABASE_NEW_USER_PASSWORD**: Provide the password for the owner of the CA Jarvis Dashboard database.
 - vii. **LDDS_DATABASE_NAME**: Provide a name for the CA Jarvis Dashboard database.
- c. Provide the information regarding the Operating system user:
- i. **OS_USER_USERNAME**: Provide user name for a non-root user. PostgreSQL does not allow a root user to do the configurations. To configure PostgreSQL database, we are creating non root user and switching to that user during the installation.
- d. Provide the Email configuration details:
- i. **EMAIL_HOST**: SMTP Host where the email server is configured. Provide valid email host to get email when the forgot password option is selected.
 - ii. **EMAIL_PORT**: Port on the host where the email server is configured.
 - iii. **EMAIL_USERNAME**: Provide the email address for the user for whom the email is configured.
 - iv. **EMAIL_PASSWORD**: Provide password to access the email account.
 - v. **EMAIL_FROMNAME**: Provide a label to identify the sender of the email.
 - vi. **EMAIL_HOSTNAME**:
- e. Provide the SSL-related information:
- i. **SSL_ENABLED** : Change to **true, if you want to enable SSL**.
 - ii. **SSL_TWO_WAY_ENABLED**: Keep **false**, if you want to enable one-way SSL. Change to **true**, if you want to enable two-way SSL.
 - iii. **SSL_PORT**: Provide the HTTPS port number.
 - iv. **KEYSTORE_FILEPATH**: Path where the keystore.jks file is located.
 - v. **TRUSTSTORE_FILEPATH**: Path where the truststore.jks file is located.
 - vi. **KEYSTORE_PASSWORD**: The password that is used to access the keystore file when SSL is enabled. Use the same password that you used while you have created the certificates.
 - vii. **TRUSTSTORE_PASSWORD**: The password that is used to access the truststore file when SSL is enabled. Use the same password that you used while you have created the certificates.
9. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes that are made to the lddsinstaller.properties file.

10. Run the following commands to install the CA Jarvis Dashboard:

```
./setup.bin -f <path>/lddsinstaller.properties
```

Example: `./setup.bin -f lddsinstaller.properties`

The installer displays the progress of installation.

The CA Jarvis Dashboard is installed regarding the configurations in the lddsinstaller.properties file. On successful installation, the installer starts all the installed services. If the services are not up in node1, manually start the services.



If you get a message saying that port is already in use, change the port number in the lddsinstaller.properties file and try to install again.

11. **(Optional)** Go to the following directory to see the log files that are generated by the installer:

```
<USER_INSTALL_DIR>/logs
```

12. Run the following commands to validate whether the servers are up and running or not:

```
netstat -an | grep <TOMCAT_PORT>
```

```
netstat -an | grep <POSTGRESQL_PORT_NODE1>
```

Proceed with the next step if both the servers are up and running.

13. Do the following steps on Node2:

- a. Repeat steps 1-4 on Node2.
- b. Copy the modified lddsInstaller.properties file from node1 to the CALDDS-Installer directory in node2.



Instead of copying the lddsInstaller.properties file from node1, you can modify the lddsInstaller.properties file in node2 as we did for node1.

- c. Repeat step 10 on node2.
- d. Run the following command to validate whether the server is up and running or not:

```
netstat -an | grep <POSTGRESQL_PORT_NODE2>
```

Installed CA Jarvis Dashboard. What Next?

Refer the following article to onboard product and tenant.

API Reference of CA Jarvis Dashboard.

After onboarding product and tenant, you are ready to use CA Jarvis Dashboard.

Installation of CA Jarvis Dashboard on a Single Node: Couchdb (Without High-Availability Failover)

Why Should I Install CA Jarvis Dashboard?

CA Jarvis Dashboard helps you visualize real-time analytics by creating comprehensive business reports. CA Jarvis Dashboard is a visualization platform that is designed to search, view, and interact with the data that is stored in CA Jarvis. So, to start visualizing the data that is stored in CA Jarvis, you should install CA Jarvis Dashboard.

Prerequisites

Software and Hardware Requirements

Platform	Memory	Processor	Hard Disk	Others
<ul style="list-style-type: none"> ▪ RHEL 7.2 ▪ CentOS 7.2 or above 	4 GB	2-Core	Hard disk: Minimum of 100-GB space	<ul style="list-style-type: none"> ▪ Unzip utility. ▪ Ensure that the following packages are installed: <ul style="list-style-type: none"> ▪ libicu-50.1.2-15.el7.x86_64 ▪ libicu-devel-50.1.2-15.el7.x86_64 ▪ libicu-doc-50.1.2-15.el7.noarch ▪ js-devel-1.8.5-19.el7.x86_64 ▪ libicu-devel-50.1.2-15.el7.x86_64 ▪ libtool-2.4.2-22.el7_3.x86_64 <p>Use the following command to install these packages:</p> <pre>yum install libicu* js-devel-1.8.5 libicu-devel libtool -y</pre>

(Optional) SSL Requirements

To enable SSL, ensure that you have the following values after setting up the SSL certificates:

- Location where keystore.jks file and truststore.jks file are stored
- The keystore and truststore passwords
- Fully Qualified Domain Name (FQDN)

Other Requirements

- Ensure that the **/etc/hosts** in all your servers have the correct value set in the following format:
 <IP Address> <Host Name with domain> <Alias>
Example:10.131.58.124 [user05-l193354.ca.com](#) user05-l193354

- Ensure that your firewall does not block CA Jarvis Dashboard from the ports it requires. Ensure that the following ports are open on the node where you are installing CA Jarvis Dashboard:
Apache Tomcat: **8080** (SSL Disabled) and **8443** (SSL Enabled.)
- Collect the IP address and Subnetmask address for the node.

Installation Instructions

The following procedure explains the steps to install CA Jarvis Dashboard on a single node.

1. Log in as a root user.
2. Download latest version of the CA Jarvis Dashboard installer package (.zip file) from CA Support site.
3. Copy the .zip file to the target machine. For example, copy the file to the */tmp* folder in the target machine.
4. Run the following command to unzip the file. Ensure that you use the correct name for the .zip file.

```
cd /tmp; unzip CALDDS-<version>.zip
```

This command creates */<location of the zip file>/CALDDS-Installer*, which contains the installation files.

5. **(This step is required if you are using a directory outside of <username>'s home directory to set values for *USER_INSTALL_DIR*.)** Ensure that <username> has the read/write permissions to the directories that are going to be used during the installation.
6. Run the following command:

```
cd CALDDS-Installer
```
7. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi lddsinstaller.properties
```
8. Make the following changes in the **lddsinstaller.properties** file:



Because this file contains passwords, ensure that you have elevated permissions for this file.

If you are editing a commented value, for example, `#USER_INSTALL_DIR`, ensure that you uncomment (remove #) while editing.

No additional spaces are allowed after the values entered in the **lddsinstaller.properties** file. For example, there should not be any spaces after the Value Y when you set the following parameter:

```
SINGLE_MACHINE=Y
```

- a. Provide the following information:
 - i. **SINGLE_MACHINE**: Keep **Y**.
 - ii. **USER_INSTALL_DIR**: Provide the location where CA Jarvis Dashboard should be installed.
- b. Provide the information regarding application server:
 - i. **APPSERVER_SERVER_PORT**: Provide the port number for your application server.
 - ii. **APPSERVER_SHUTDOWN_PORT**: Provide the shutdown port number for your application server.
- c. Provide the information regarding CouchDB:
 - i. **DATABASE_PORT**: Provide the port number for CouchDB.
 - ii. **DATABASE_NEW_USER_USERNAME**: Provide the CouchDB user name. This user will be the owner of the CA Jarvis Dashboard database.
 - iii. **DATABASE_NEW_USER_PASSWORD**: Provide the password for the owner of CA Jarvis Dashboard database.
- d. Provide the information regarding the Operating system user:
 - i. **OS_USER_USERNAME**: Provide user name for a non-root user. During installation, the root user will be switched to this non-root user.
- e. Provide the Email configuration details:
 - i. **EMAIL_HOST**: SMTP Host where the email server is configured. Provide valid email host to get email when the forgot password option is selected.
 - ii. **EMAIL_PORT**: Port on the host where the email server is configured.
 - iii. **EMAIL_USERNAME**: Provide the email address for the user for whom the email is configured.
 - iv. **EMAIL_PASSWORD**: Provide password to access the email account.
 - v. **EMAIL_FROMNAME**: Provide a label to identify the sender of the email.
 - vi. **EMAIL_HOSTNAME**: Provide the host name where **CA Jarvis Dashboard** is installed.
- f. Provide the SSL-related information:
 - i. **SSL_ENABLED** : Change to **true**, if you want to enable SSL.
 - ii. **SSL_TWO_WAY_ENABLED**: Keep **false**, if you want to enable one-way SSL. Change to **true**, if you want to enable two-way SSL.
 - iii. **SSL_PORT**: Provide the HTTPS port number.
 - iv. **KEYSTORE_FILEPATH**: Path where the keystore.jks file is located.

- v. **TRUSTSTORE_FILEPATH**: Path where the truststore.jks file is located.
 - vi. **KEYSTORE_PASSWORD**: The password that is used to access the keystore file when SSL is enabled. Use the same password that you used while you have created the certificates.
 - vii. **TRUSTSTORE_PASSWORD**: The password that is used to access the truststore file when SSL is enabled. Use the same password that you used while you have created the certificates.
9. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes that are made to the lddsinstaller.properties file.
 10. Run the following commands to install CA Jarvis Dashboard:

```
./setup.bin -f <path>/lddsinstaller.properties
```

Example: `./setup.bin -f lddsinstaller.properties`

The installer displays the progress of installation.

CA Jarvis Dashboard is installed as per the configurations in the lddsinstaller.properties file. The On successful installation, the installer starts all the installed services.



If you get a message saying that port is already in use, change the port number in the lddsinstaller.properties file and try to install again.

11. **(Optional)** Go to the following directory to see the log files that are generated by the installer:

```
<USER_INSTALL_DIR>/logs
```



If the installation fails, refer the `/tmp` directory for logs.

12. **(Optional:)** Change to `<USER_INSTALL_DIR>/bin` and execute the following commands to start or stop the CA Jarvis Dashboard services. Remember that the installer starts all the CA Jarvis Dashboard services after the installation.

To start the Tomcat and CouchDB database:

```
./startservers.sh
```

To stop the Tomcat and CouchDB database:

```
./stopservers.sh
```

Installed CA Jarvis Dashboard What Next?

Refer the following article to onboard product and tenant.

[API Reference of CA Jarvis Dashboard.](#)

After onboarding product and tenant, you are ready to use CA Jarvis Dashboard.

Upgrading and Migrating

- [Upgrading](#)
- [Migrating and Exporting](#)

Upgrading

- [Upgrade CA Jarvis from 2.2 to 2.3](#)

Upgrade CA Jarvis from 2.2 to 2.3

This document helps you upgrade CA Jarvis from release 2.0.4 and 2.1.0 to 2.2.

- [\(Single Node\) Upgrade CA Jarvis from 2.2 to 2.3](#)
- [\(Multi-Node\) Upgrade CA Jarvis from 2.2 to 2.3](#)

(Single Node) Upgrade CA Jarvis from 2.2 to 2.3

This article helps you upgrade CA Jarvis 2.2 single-node installation to release 2.3.

Prerequisites and Recommendations

- The existing system should be in Jarvis 2.2 release with all the hotfixes applied.
- All Jarvis components are deployed on RHEL or CentOS Virtual machines.
- Refer [here](#) for the Software and Hardware requirements.

Upgrade Instructions

Review the following points before starting the upgrade:



1. There will be a downtime while upgrading the CA Jarvis components.
2. Place the following directories in a different location (outside the Elasticsearch folder) so that there is no chance of deleting them when upgrading Elasticsearch:
 - a. data
 - b. logs



This procedure upgrades both the Jarvis components and Elasticsearch.



You can find the errors that might happen during the upgrade, and the possible resolutions [here](#).

The following procedure explains the steps to upgrade CA Jarvis on a single node.

1. Log in as a root user.
2. Download latest version of the Jarvis package (.tar file) from CA Support site.
3. Copy the .tar file to the target machine. For example, copy the file to the `/tmp` folder in the target machine.

4. Run the following command to untar the file. Ensure that you use the correct name for the .tar file.

```
cd /tmp; tar -xvf jarvisInstaller_installAnywhere-<version>.tar
```

This command creates `/location of the tar file>/jarvisInstaller_installAnywhere`, which contains the upgrade files and Jarvis binaries.

5. Ignore step 5 and 6, if you have already created a non-root user, and want to use the same user for upgrading Jarvis. If want to create a non-root user for upgrading Jarvis, run the following command and continue with step 6:

```
cd jarvisInstaller_installAnywhere/scripts
```

6. Create a non-root user by running the following commands. Ensure that you customize the second command and provide password for the non-root user when prompted:

```
chmod +x prepareMachineAsRoot.sh
./prepareMachineAsRoot.sh <username>
```

<username> must be a string.

From step 10, you will be using the non-root-user credentials that you have set in this step.

7. Run the following command:

```
cd /tmp
```

8. Run the following commands one by one:

```
chown -R <username>:<username> jarvisInstaller_installAnywhere

chmod +x jarvisInstaller_installAnywhere/CA_Analytics.bin
```

9. Ensure that <username> has the read/write permissions to the directories that are going to be used during the upgrade process. This is required if you are using a directory outside of <username>'s home directory to set values for `USER_INSTALL_DIR`. To set permissions, complete the following steps:

- a. Change to the directory to which you want to set the permissions.
- b. Customize and run the following commands:


```
chmod 755 <name of the directory>
chown -R <username>:<username> <name of the directory>
```

10. Log in as the non-root user to whom the permissions were set in step 6.

```
su - <username>
```

11. Copy the `jarvisInstaller_installAnyWhere` directory to the home directory of `<username>` (`/home/<username>`):

Example:

```
cp -r /tmp/jarvisInstaller_installAnyWhere /home/<username>/.
```

12. Run the following command:

```
cd jarvisInstaller_installAnyWhere/properties
```

13. Run the following command, and press `i` (lowercase `I`) in the keyboard to change to the insert mode:

```
vi analyticsUpgrade.properties
```

14. Make the following changes in the **analyticsUpgrade.properties** file by providing the following information:

- a.
 - i.
 1. **USER_INSTALL_DIR**: Provide the absolute path to the location where Jarvis is already installed. Ensure that `<username>` has read/write permissions to this location as mentioned in step 9.
 2. **(Optional) VERIFIER_KAFKA_CONSUMER_COUNT=4**: Provide the total number of Kafka consumer in Verifier.
 3. **(Optional) INDEXER_KAFKA_CONSUMER_COUNT=4**: Provide the total number of Kafka consumer in Indexer.



Ensure that you provide valid values for **KAFKA_INSTALLATION_HOST_PORTS**, **ZOOKEEPER_INSTALLATION_HOST_PORTS**, and **SCHEMA_REGISTRY_INSTALLATION_HOSTS** to install and configure Schema Registry.

4. **KAFKA_INSTALLATION_HOST_PORTS**: Provide the IP address or FQDN and port number (9092) of the host where Kafka should be installed. If you do not provide this value, Kafka will not be installed. Use the following command to find the IP address. You can find the IP address after "inet adr" in the output.

```
/sbin/ifconfig Use the following command if the
previous command fails because of privilege-related
error: sudo /sbin/ifconfig
```

5. **ZOOKEEPER_INSTALLATION_HOST_PORTS**: Provide the IP address or FQDN and port number (2181) of the host where ZOOKEEPER should be installed.

6. **SCHEMA_REGISTRY_INSTALLATION_HOSTS**=host1:port,host2:port.



Provide the IP address or FQDN and port number (8081) of the host where schema registry should be installed. If you do not provide this value, schema registry will not be installed. There should not be spaces between host names.

7. Provide the following details if your CA Jarvis Dashboard setup is SSL-enabled:

a. **LDDS_SSL_CONFIGURED=false**. Change to **true** if CA Jarvis Dashboard is SSL-enabled.

b. **LDDS_HOST=localhost:8080**. Provide the IP address or FQDN of the CA Jarvis Dashboard host.

8. Provide the following SSL-related values, if SSL is already enabled during the installation of CA Jarvis. Ensure that you provide the same SSL values provided during installation.

a. **SSL_ENABLED**

b. **KEYSTORE_FILEPATH**

c. **TRUSTSTORE_FILEPATH**

d. **KEYSTORE_PASSWORD**

e. **TRUSTSTORE_PASSWORD**

f. **KEY_PASSWORD**

9. **KRON_HOST**: Provide the IP address or FQDN of the host where KRON is installed.

15. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes made to the analyticsUpgrade.properties file.

16. Change to the parent directory :

Example:

```
cd /home/<username>/jarvisInstaller_installAnyWhere
```

All set to upgrade CA Jarvis to the latest release.

17. Upgrade CA Jarvis by running the following command:

```
./CA_Analytics.bin -f <path>/analyticsUpgrade.properties
```

Example:

```
./CA_Analytics.bin -f properties/analyticsUpgrade.properties
```

The installer displays the progress of upgrade. The Jarvis Health-check is executed as part of the upgrade. The installer displays the location of the log file to see the progress of the execution of Jarvis Health-check.

Jarvis is upgraded as per the configurations in the **analyticsUpgrade.properties** file. The On successful upgrade, the installer starts all the Jarvis services, except the Verifier service. If Verifier service starts before APIs service comes up and finishes schema migration, Verifier fails.

18. Use the following command to know the status of Verifier:

```
systemctl status ca_verifier-ingest.service
```

If the status of Verifier is not active, you must restart Verifier by running the following command:

```
systemctl start ca_verifier-ingest.service
```

19. Start Schema Registry by running the following command:

```
systemctl start ca_schema-registry.service
```

20. Check the log messages in <User_Install_Dir>/Analytics/logs/indexer.log. If you see the following error, start Indexer:

```
java.lang.NullPointerException
```

```
2017-08-30 18:53:25 ERROR AvroDocumentIndexer:108 - Error while
deserializing record.
```

```
org.apache.kafka.common.errors.SerializationException: Unknown magic byte!
```

Command to start Indexer:

```
systemctl start ca_indexer-ingest.service
```

The upgrade process has been completed successfully.

(Multi-Node) Upgrade CA Jarvis from 2.2 to 2.3

This article helps you upgrade CA Jarvis 2.2 multi-node installation to release 2.3.

Prerequisites and Recommendations

- The existing system should be in Jarvis 2.2 release with all the hotfixes applied.
- All Jarvis components are deployed on RHEL or CentOS Virtual machines.
- Refer [here](#) for the Software and Hardware requirements.

Upgrade Instructions

Review the following points before starting the upgrade:



1. There will be a downtime while upgrading the CA Jarvis components.
2. Place the following directories in a different location (outside the Elasticsearch folder) so that there is no chance of deleting them when upgrading Elasticsearch:
 - a. data
 - b. logs



This procedure upgrades both the Jarvis components and Elasticsearch.



You can find the errors that might happen during the upgrade, and the possible resolutions [here](#).

The following procedure explains the steps to upgrade CA Jarvis on multiple nodes.

1. Log in as a root user.
2. Download latest version of the Jarvis package (.tar file) from CA Support site.
3. Copy the .tar file to the target machine. For example, copy the file to the `/tmp` folder in the target machine.
4. Run the following command to untar the file. Ensure that you use the correct name for the .tar file.

```
cd /tmp; tar -xvf jarvisInstaller_installAnywhere-<version>.tar
```

This command creates `/<location of the tar file>/jarvisInstaller_installAnywhere`, which contains the upgrade files and Jarvis binaries.

5. Ignore step 5 and 6, if you have already created a non-root user, and want to use the same user for upgrading Jarvis. If want to create a non-root user for upgrading Jarvis, run the following command and continue with step 6:

```
cd jarvisInstaller_installAnywhere/scripts
```

6. Create a non-root user by running the following commands. Ensure that you customize the second command and provide password for the non-root user when prompted:

```
chmod +x prepareMachineAsRoot.sh
./prepareMachineAsRoot.sh <username>
```

<username> must be a string.

From step 10, you will be using the non-root-user credentials that you have set in this step.

7. Run the following command:

```
cd /tmp
```

8. Run the following commands one by one:

```
chown -R <username>:<username> jarvisInstaller_installAnywhere
```

```
chmod +x jarvisInstaller_installAnywhere/CA_Analytics.bin
```

9. Ensure that <username> has the read/write permissions to the directories that are going to be used during the upgrade process. This is required if you are using a directory outside of <username>'s home directory to set values for USER_INSTALL_DIR. To set permissions, complete the following steps:

a. Change to the directory to which you want to set the permissions.

b. Customize and run the following commands:

```
chmod 755 <name of the directory>
```

```
chown -R <username>:<username> <name of the directory>
```

10. Log in as the non-root user to whom the permissions were set in step 6.

```
su - <username>
```

11. Copy the jarvisInstaller_installAnywhere directory to the home directory of <username> (/home/<username>) :

Example:

```
cp -r /tmp/jarvisInstaller_installAnywhere /home/<username>/.
```

12. Run the following command:

```
cd jarvisInstaller_installAnywhere/properties
```

13. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi analyticsUpgrade.properties
```

14. Make the following changes in the **analyticsUpgrade.properties** file by providing the following information:

- a.
 - i.
 1. **USER_INSTALL_DIR**: Provide the absolute path to the location where Jarvis is already installed. Ensure that <username> has read/write permissions to this location as mentioned in step 9.
 2. **(Optional) VERIFIER_KAFKA_CONSUMER_COUNT=4**: Provide the total number of Kafka consumer in Verifier.
 3. **l(Optional) INDEXER_KAFKA_CONSUMER_COUNT=4**: Provide the total number of Kafka consumer in Indexer.



Ensure that you provide valid values for **KAFKA_INSTALLATION_HOST_PORTS**, **ZOOKEEPER_INSTALLATION_HOST_PORTS**, and **SCHEMA_REGISTRY_INSTALLATION_HOSTS** to install and configure Schema Registry.

4. **KAFKA_INSTALLATION_HOST_PORTS**: Provide the IP address or FQDN and port number (9092) of the host where Kafka should be installed. If you do not provide this value, Kafka will not be installed. Use the following command to find the IP address. You can find the IP address after "inet adr" in the output.

```
/sbin/ifconfig Use the following command if the
previous command fails because of privilege-related
error: sudo /sbin/ifconfig
```

5. **ZOOKEEPER_INSTALLATION_HOST_PORTS**: Provide the IP address or FQDN and port number (2181) of the host where ZOOKEEPER should be installed.
6. **SCHEMA_REGISTRY_INSTALLATION_HOSTS**=host1:port,host2:port.



Provide the IP address or FQDN and port number (8081) of the host where schema registry should be installed. If you do not provide this value, schema registry will not be installed. There should not be spaces between host names.

7. Provide the following details if your CA Jarvis Dashboard setup is SSL-enabled:
 - a. **LDDS_SSL_CONFIGURED=false**. Change to **true** if CA Jarvis Dashboard is SSL-enabled.
 - b. **LDDS_HOST=localhost:8080**. Provide the IP address or FQDN of the CA Jarvis Dashboard host.
8. Provide the following SSL-related values, if SSL is already enabled during the installation of CA Jarvis. Ensure that you provide the same SSL values provided during installation.
 - a. **SSL_ENABLED**
 - b. **KEYSTORE_FILEPATH**
 - c. **TRUSTSTORE_FILEPATH**
 - d. **KEYSTORE_PASSWORD**
 - e. **TRUSTSTORE_PASSWORD**
 - f. **KEY_PASSWORD**

9. **KRON_HOST**: Provide the IP address or FQDN of the host where KRON is installed.

15. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes made to the analyticsUpgrade.properties file.

16. Change to the parent directory :

Example:

```
cd /home/<username>/jarvisInstaller_installAnywhere
```

All set to start the upgrade.



Run the upgrade command on the Elasticsearch nodes one-by-one. After upgrading the Elasticsearch nodes, upgrade the nodes where you have installed CA Jarvis services.

17. Upgrade all the Elasticsearch nodes first:

- a. Do the rolling upgrade of Elasticsearch to 5.5.3 on a node where you have installed Elasticsearch by running the following command:



Elasticsearch cluster should be upgraded one node at a time so that there will not be any downtime.

Ensure that all the nodes in the cluster are upgraded one-after-another without having long delay.

The following command upgrades both Elasticsearch and other CA Jarvis services installed on the selected node.

```
./CA_Analytics.bin -f <path>/analyticsUpgrade.properties
```

Example:

```
./CA_Analytics.bin -f properties/analyticsUpgrade.properties
```

Elasticsearch is upgraded to 5.5.3 on that node. If the node has any other CA Jarvis CA Jarvis services installed on it, the installer upgrades those services also.

- b. Customize and run the following command to confirm that the cluster is stable and the node has recovered after the upgrade.

```
curl -XGET '<host name>:9200/_cat/health?pretty'
```

- c. Repeat steps 17.a and 17.b on all remaining Elasticsearch nodes.
You have upgraded Elasticsearch on all the nodes where you have installed Elasticsearch. You can now upgrade the CA Jarvis services.

18. Upgrade the nodes where you have installed CA Jarvis services (that is, non-Elasticsearch nodes) :

- a. Run the following commands to stop Verifier (to stop processing any new documents till the upgrade is completed) and API services (to stop accepting new ingestion):

```
systemctl stop ca_verifier-ingest.service
systemctl stop ca_Tomcat.service
```

- b. Run the following commands to upgrade CA Jarvis services:

```
./CA_Analytics.bin -f <path>/analyticsUpgrade.properties
```

Example:

```
./CA_Analytics.bin -f properties/analyticsUpgrade.properties
```

The installer displays the progress of upgrade. You have to [manually run the Jarvis Health-check](#) to know the health of installation.

Jarvis is upgraded as per the configurations in the **analyticsUpgrade.properties** file. On successful upgrade, the installer starts all the Jarvis services, except the Verifier service. If Verifier service starts before APIs service comes up and finishes schema migration, Verifier fails.

- c. Use the following command to know the status of Verifier:

```
systemctl status ca_verifier-ingest.service
```

If the status of Verifier is not active, you must restart Verifier by running the following command:

```
systemctl start ca_verifier-ingest.service
```

- d. Start Schema Registry by running the following command:

```
systemctl start ca_schema-registry.service
```

- e. Check the log messages in <User_Install_Dir>/Analytics/logs/indexer.log. If you see the following error, start Indexer:

```
java.lang.NullPointerException
```

```
2017-08-30 18:53:25 ERROR AvroDocumentIndexer:108 - Error while
deserializing record.
```

```
org.apache.kafka.common.errors.SerializationException: Unknown magic byte!
```

Command to start Indexer:

```
systemctl start ca_indexer-ingest.service
```

The upgrade process has been completed successfully.

Migrating and Exporting

The following topics assist with the migration and exporting of CA Jarvis InstallAnywhere to various platforms.

- [Export CA Jarvis Data](#)

Export CA Jarvis Data

The migration tool helps you export data stored in CA Jarvis as a zip file.

The following topic provides instructions on exporting data stored in CA Jarvis as a zip file. This is a common procedure for InstallAnywhere and Docker-based installations of CA Jarvis.

- [Prerequisites](#)
- [Export Data from CA Jarvis](#)

Prerequisites

To ensure a successful exporting, verify that your system meets the prerequisites.

- CA Jarvis 2.3
- The location where you want the zip file to be created must have enough storage space with required read/write permissions.

Export Data from CA Jarvis

This procedure helps you export data stored in CA Jarvis as a zip file.

Follow these steps,

1. Download the migration.zip file.
2. Unzip the migration.zip file.
3. Update the export.properties file:
 - a. **productId**: Enter the identifier of the product from which you want to export the data.
 - b. **tenantId**: Enter the identifier of the tenant from which you want to export the data.
 - c. **(Optional) docTypeId**: Enter the identifier of the document type from which you want to export the data. Use only if you want to export for specific docType. If you want to export data for all docTypes for a tenant, do not specify docTypeId in the properties file.
 - d. **(Optional) docTypeVersion**: Enter the doctype version from which you want to export the data.
 - e. **zipLocation**: Enter the location where you want to keep the exported zip file.
 - f. **startTime** (Optional). Specify the start date and time from which the data should be exported. Use yyyy-MM-dd HH:mm:ss.SSS as the format.
 - g. **endTime**: (Optional). Specify the end date and time to which the data should be exported. Use yyyy-MM-dd HH:mm:ss.SSS as the format.

h. **(Optional)** Provide SSL-related values, if you want to enable SSL:

- i. **trustStoreFilePath**: Path where the truststore.jks file is located.
- ii. **trustStorePassword**: The password that is used to access the truststore file when SSL is enabled. Use the same password that you used while you have created the certificates.
- iii. **keyStoreFilePath**: Path where the keystore.jks file is located.
- iv. **keyStorePassword**: The password that is used to access the keystore file when SSL is enabled. Use the same password that you used while you have created the certificates.

i. **url**: URL of the Jarvis instance from which you need to export data.

4. Ensure that CA Jarvis is running, then use one of the following commands to run the migration tool and to export the data:

```
java -jar migration-tools.jar -m export -props <path to export.properties>
```

Example:

```
java -jar migration-tools.jar -m export -props /home/jarvis/migration-tools/export.properties
```

Configuring

- [Secure Sockets Layer \(SSL\)](#)
- [Capture Usage Metrics Using Jmetric](#)
- [Back Up and Restore Elasticsearch Data](#)

Secure Sockets Layer (SSL)

- [Set Up SSL Certificates](#)

Set Up SSL Certificates

- [Set Up SSL Certificates Obtained From Certificate Authority \(Recommended in Production Environment\)](#)
 - [Prerequisite:](#)
- [Set Up Self-Signed Certificates \(Recommended only for Testing Purpose in Non-Production Environment\)](#)
 - [Moving to CA-Signed Certificates from Self-Signed Certificates](#)

Set Up SSL Certificates Obtained From Certificate Authority (Recommended in Production Environment)

For securing the communication channel among various components using Secure Socket Layer (SSL) authentication, set up the SSL certificates. This topic helps you set up SSL Certificates for Apache Tomcat and for Jarvis API services.

Prerequisite:

- Ensure that you have Java™ Platform, Standard Edition Development Kit (JDK™) installed on your server. This procedure uses the Java Keytool utility for generating keys and for setting up digital certificate. Keytool is a key and certificate management utility, which is part of JDK.
- You must be a root user or a user with root permissions.



Provide only FQDN (not IP address) while setting up SSL certificates. FQDN is case-sensitive.



Note down the the following values after setting up the SSL certificates. You must provide these values during the installation as the value of KEYSTORE_FILEPATH, TRUSTSTORE_FILEPATH, KEYSTORE_PASSWORD, TRUSTSTORE_PASSWORD, and KEY_PASSWORD.

- Location where keystore.jks file and truststore.jks file are stored

- The keystore and truststore passwords
- Fully-Qualified Domain Name (FQDN)

1. Change to the same directory where Keytool resides (/jre/bin/keytool), and use the following commands to generate keystore for both the Tomcat server and the API services:

```
keytool -keystore <keystore path> -alias <alias name of the key> -validity <validity> -genkey
```

Example:

```
keytool -keystore my-server.keystore.jks -alias my-server -validity 365 -genkey
```

Note:

The Common Name (CN) should be the host name and is case-sensitive.
 These commands prompt for the following:
 common name: Set as hostname<machinename>.
 Password: set your password.
 Also, fill the other details.

2. Create a Certificate Signing Request (CSR) :

```
keytool -certreq -alias <alias name of the key> -keystore <keystore path> -storepass <keystore password> -file <name of the CSR file>
```

Example:

```
keytool -certreq -alias my-server -keystore my-server.keystore.jks -storepass <keystore password> -file CSR-for-my-server
```

3. Get the digital certificate from an authorized Certificate Authority (CA). For example, VeriSign, Thawte, or Godaddy.
4. Trust the digital certificate for both Tomcat server and API clients:

- a. Run the following command for both the server and client one-by-one, type the password twice, and type "yes" to trust the digital certificate:

```
keytool -keystore <truststore path> -alias <alias name of the certificate> -import -file <name of the digital certificate>
```

Example:

```
keytool -keystore my-server.truststore.jks -alias CARoot -import -file certificate-for-my-server
```

5. Complete the following sub-steps:

- a. Run the following command to export the server and client certificates:

```
keytool -keystore <keystorepath> -alias <alias name of the key> -certreq -file <name of the exported key>
```

Note: Ensure that you give the same alias that was given in Step 1 for the Tomcat server and API client.

Example:

```
keytool -keystore my-server.keystore.jks -alias my-server -certreq -file my-server-cert-file
```

- b. Get the exported key cryptographically signed by the CA.

Example:

```
openssl x509 -req -CA ca-cert -CAkey ca-key -in my-server-cert-file -out my-server-cert-signed -days 365 -CAcreateserial -passin pass:changeit
```

- c. Run the following commands to import the digital certificate and the signed certificates:

```
keytool -keystore <path to the keystore> -alias <alias name of the certificate> -import -file <name of the digital certificate>
keytool -keystore <path to the keystore> -alias <alias name of the key> -import -file <name of the signed key>
```

Example:

```
keytool -keystore my-server.keystore.jks -alias CARoot -import -file certificate-for-my-server
keytool -keystore my-server.keystore.jks -alias my-server -import -file my-server-cert-signed
```

- d. Run the following command to import the digital certificate and the signed certificates into JDK:

This command assumes that your Java_Home environment variable is already set.

```
keytool -keystore <$JAVA_HOME/..jre/lib/security/cacerts> -alias <alias name of the certificate> -import -file <name of the digital certificate>
```

Example:

```
keytool -keystore $JAVA_HOME/..jre/lib/security/cacerts -alias CARoot -import -file certificate-for-my-server
```

- e. Run the following command to remove the cacert from the Keystore:

```
keytool -keystore <path to the keystore> -alias <alias name of the certificate> -delete
```

Example:

```
keytool -keystore my-server.keystore.jks -alias CARoot -delete
```

Set Up Self-Signed Certificates (Recommended only for Testing Purpose in Non-Production Environment)

A self-signed certificate is signed by the person who creates it rather than by a trusted certificate authority. So, it is not recommended for your production environment. Use this procedure only for testing the functionality of SSL in non-production environment.



Note down the directory path where you are running the script and the password. You must provide this path and password during the installation as the value of KEYSTORE_FILEPATH, TRUSTSTORE_FILEPATH, KEYSTORE_PASSWORD, TRUSTSTORE_PASSWORD, and KEY_PASSWORD.

The scripts that are mentioned in the following procedures are available in the `\JarvisInstaller_installAnywhere\scripts` directory of the Jarvis installer package.

On Single Node:

1. Download generate_dev_certs.sh.
2. Run one of the following commands to generate the certificates:
 - a. Run the following command, if you want to use the default password (**jarvis@123**):

```
./generate_dev_certs.sh
```

- b. Run the following command, if you want to use your own password:

```
./generate_dev_certs.sh <your password>
```

On Multiple Nodes:

You must generate the certificates on one random machine.

1. Download the following scripts:

a. generate_dev_certs.sh

b. generate_dev_certs_cluster.sh

2. Run one of the following commands on any of the machines to generate the certificates:

- a. Run the following command, if you want to use the default password (**jarvis@123**):

```
./generate_dev_certs.sh
```

- b. Run the following command, if you want to use your own password:

```
./generate_dev_certs.sh <your password>
```

3. Copy the **ca-cert** and **ca-key** files to all other machines in your cluster.

4. Run one of the following commands on all other machines in your cluster:

- a. Run the following command, if you want to use the default password (**jarvis@123**):

```
./generate_dev_certs_cluster.sh ca-cert ca-key
```

- b. Run the following command, if you want to use your own password:

```
./generate_dev_certs_cluster.sh ca-cert ca-key <your password>
```

Moving to CA-Signed Certificates from Self-Signed Certificates

You may have used self-signed certificates to enable SSL, and later decided to move to CA-signed certificates to ensure more security. In CA-signed certificates, the file names and passwords may or may not be the same. The following table summarizes the different scenarios on moving to CA-signed certificates from self-signed certificates:

Scenario	Action
File names and passwords are the same and the files are present in the configured location	None
File names and passwords are the same, but the files are not present in the configured location	Copy the files to the configured location. Example: Indexer.properties configuration keystore_filepath=/opt/sslcert/keystore.jks If the new certificates are in the /opt directory, then copy the new certificates to the /opt/sslcert/ folder.

Scenario	Action
File names, paths, or passwords are different for the CA-signed certificates.	Do the following procedure.
	<ol style="list-style-type: none"> 1. Stop all CA Jarvis services by running the following command. <code>systemctl stop ca_*</code> 2. Customize and run the following command: <code>cd <USER_INSTALL_DIR>/config</code> 3. Update keystore_password or truststore_password if the passwords are changed. Update keystore_filepath or truststore_filepath if the file is stored in a different location or the file name is changed. In both the cases, you should update and save the values in the following files: <ol style="list-style-type: none"> a. apis.properties b. Indexer.properties c. verifier.properties d. utils.properties e. kron.properties 4. (Applicable only if Tomcat is installed on the node). Do the following steps: <ol style="list-style-type: none"> a. Customize and run the following command: <code>cd <USER_INSTALL_DIR>/apache-tomcat-<version>/conf</code> b. Open Server.xml. c. Search for keystoreFile, keystorePass, truststoreFile, and truststorePass, and update with the new file path and passwords. 5. (Applicable only if Kafka is installed on the node). Do the following steps: <ol style="list-style-type: none"> a. Customize and run the following command: <code>cd <USER_INSTALL_DIR>/kafka-<version>/config</code> b. Open server.properties. c. Search for save ssl.keystore.location, ssl.keystore.password, and ssl.key.password and update with the new file path and passwords. 6. (Applicable only if ElasticSearch is installed on the node). Do the following steps: <ol style="list-style-type: none"> a. Customize and run the following command: <code>cd <USER_INSTALL_DIR>/elasticsearch-<version>/config</code> b. Open elasticsearch.yml. c. Search for the following parameters, and update with the new file path and passwords:

- i. searchguard.ssl.transport.keystore_password: <keystorePassword>
 - ii. searchguard.ssl.transport.truststore_password: <truststorePassword>
 - iii. searchguard.ssl.transport.truststore_filepath: <relative path under config /truststore.jks>
 - iv. searchguard.ssl.transport.keystore_filepath: <relative path under config /keystore.jks>
 - v. searchguard.ssl.http.keystore_password: <keystorePassword>
 - vi. searchguard.ssl.http.truststore_password: <truststorePassword>
 - vii. searchguard.ssl.http.truststore_filepath: <relative path under config /truststore.jks>
 - viii. searchguard.ssl.http.keystore_filepath: <relative path under config /keystore.jks>
- d. Copy the new truststore and keystore files to the <USER_INSTALL_DIR> /elasticsearch_<version>/config/<sslcertspath> folder.



Refer elasticsearch.yml to know the current path. The path is relative to the config directory.

7. Start all CA Jarvis services by running the following command.

```
systemctl start ca_*
```

Your CA Jarvis setup now has the protection of CA-signed certificates.

Reload SSL Certificates

This procedure assumes that you have already generated the SSL certificates as mentioned in Set Up SSL Certificates, and you have enabled SSL on your Jarvis setup. This procedure describes what you should do if your SSL certificates are expired or if you have lost the certificates, and want to load new certificates.



This procedure uses same path names, alias names, and so on that are used in Set Up SSL Certificates for consistency.

To see the list of certificates that are present in your Keystore or Truststore, use the following command:

```
keytool -keystore <path to the keystore or truststore> -list -v
```

1. Customize and run the following command to delete the ca certificates from the Truststore.


```
keytool -keystore <path to the truststore> -alias <alias name of the ca certificate> -delete
```

Example:

```
keytool -keystore my-server.truststore.jks -alias CARoot -delete
```

2. Customize and run the following command to delete the signed certificates from the Keystore.

```
keytool -keystore <path to the keystore> -alias <alias name of the signed certificate> -delete
```

Example:

```
keytool -keystore my-server.keystore.jks -alias my-server -delete
```

3. Repeat the steps in Set Up SSL Certificates on your nodes.
4. Do the following steps on the nodes where you have installed Elasticsearch:
 - a. Open the directory where you have installed ElasticSearch.
 - b. Open the Config/elasticsearch.yml file and update the modified Trustore and Keystore paths there.

5. Restart all the Jarvis services:

```
sudo systemctl start <serviceName>
```

Example:

```
sudo systemctl start ca_elasticsearch.service
```

Refer the [Troubleshooting section](#) for the complete list of services.

Capture Usage Metrics Using Jmetric

A Jarvis admin would like to have insights on how the customers are using the product. For example, you would like to get key metrics across tenants, within a tenant, or across products. These metrics help you evaluate the system usage and suggest improvements based on customer usage patterns and behavior. We use Jmetric to capture insights on how the customers are using the product and how Jarvis system is performing at system level. We onboard Jmetric doc type by default for new products. This doc type captures various metrics from different sources (Ingestion, DAS, Visualization, and System). You need to query the Jmetric to get the captured insights. CA LDDs also can visualize these insights.

Following table summarizes the different metrics Jmetric captures. Jarvis captures these metrics at specific intervals, which is set for each metric level. If interval is not set for any metric, Jarvis uses the default interval time (1 minute).

Source	Metric	Interval	Scope	Description
Onboarding	products_onboarded	30 seconds	Jarvis level	Number of products onboarded.
	tenants_onboarded	30 seconds	Individual product-tenant level	Number of tenants onboarded.
	doctypes_onboarded			

Source	Metric	Interval	Scope	Description
Ingestion		30 seconds	Individual product-tenant level	Number of doc type onboarded.
	doc_ingested	60 seconds	Individual product-tenant level	Number of documents.
	docsize_ingested	60 seconds	Individual product-tenant level	Size of the document.
	payload_ingested	60 seconds	Product Level	Number of payloads ingestion (basically number of Ingestion API calls)
	payload_priority	60 seconds	Product Level	Number of payloads ingested based on priority.
	payload_size	60 seconds	Product Level	Maximum, minimum, and average size of payloads that are ingested.
	documents_failed_ingestion	60 seconds	Individual product-tenant level	Number of failed documents due to invalid date formats.
Indexer	doc_indexed	60 seconds	Individual product-tenant level	Number of documents ingested at the indexer level.
	doc_indexed_failed	60 seconds	Jarvis level	Number of documents failed to ingest.
DAS	das_query_count	60 seconds	Individual product-tenant level	Number of DAS queries.
	das_response_time		Individual product-tenant level	Response time of the DAS query.
Viewlets (CA LDDS)	viewlets_created	Event based	Individual product-tenant level	Number of viewlets created.
	viewlet_viewd	60 seconds	Individual product-tenant level	Number of times the viewlet are viewed.
Dashboard (CA LDDS)	dashboard_created	Event based	Individual product-tenant level	Number of dashboard created.
	dashboard_viewed	60 seconds	Individual product-tenant level	Number of times the dashboard is viewed.
	user_created	Event based	Individual product-tenant level	Number of LDDS users created.
Elasticsearch	free_in_bytes	60 seconds	Jarvis level	Total number of unallocated bytes in the file store.
	read_operations		Jarvis level	

Source	Metric	Interval	Scope	Description
		60 seconds		Total number of read operations for the device completed since Elasticsearch is started.
	write_operations	60 seconds	Jarvis level	Total number of write operations for the device completed since Elasticsearch is started.
	index_count	60 seconds	Jarvis level	Total number of indexes.
	shard_count	60 seconds	Jarvis level	Total number of shards.
	indices_size_bytes	60 seconds	Jarvis level	Cluster store size in bytes.
	cpu_usage	60 seconds	Jarvis level	Recent CPU usage for the system.
	memory_usage	60 seconds	Jarvis level	Percentage of used memory.
Kafka	MessagesInPerSec	60 seconds	Jarvis level	Aggregate incoming message rate.
	ConsumerLag/MaxLag	60 seconds	Jarvis level	Number of messages consumer is behind producer / Maximum number of messages consumer is behind producer.
	PartitionCount	60 seconds	Jarvis level	Number of partitions across all brokers.
	request-rate	60 seconds	Jarvis level	The average number of requests sent per second.
	response-rate	60 seconds	Jarvis level	The average number of responses received per second.
	request_latency_avg	60 seconds	Jarvis level	The average request latency in milliseconds.
	records_per_request_avg	60 seconds	Jarvis level	The average number of records in each request.
	records_consumed_rate	60 seconds	Jarvis level	The average number of records consumed per second.
Zookeeper	fetch_rate	60 seconds	Jarvis level	The number of fetch requests per second.
	request_latency_avg		Jarvis level	Response time of the server towards a client request (since the server was started).
Spark	spark_jobs		Jarvis level	Number of spark jobs that are executing.
Hadoop	dfs_capacity_used		Jarvis level	

Source	Metric	Interval	Scope	Description
Kron				Storage space used by HDFS files in bytes.
	files_total		Jarvis level	Number of files and directories in the HDFS.
	jobs_executed	1 minute	Jarvis level	Number of Kron jobs that are running
	jobs_failed	1 minute	Jarvis level	Number of Kron jobs that are failed.
Verifier	payload_ingested	1 minute	Product Level	Number of payloads ingestion (basically number of Ingestion API calls)
	payload_priority	1 minute	Product Level	Number of payloads ingested based on priority.
	payload_size	1 minute	Product Level	Maximum, minimum and average size of payloads ingested.
	documents_failed_ingestion	1 minute	Individual product-tenant level	Number of failed documents due to header invalidation.
	doc_ingested	1 minute	Individual product-tenant level	Number of (valid) documents that are ingested.
	docsize_ingested	1 minute	Individual product-tenant level	Size of (valid) documents that are ingested.

Sample JMetrics DocType Schema

```
{
  "product_id": "",
  "doc_type_id": "jmetrics",
  "doc_type_version": "1.0",
  "tenant_id": "",
  "unique_id": "${'metricid'}",
  "mappings": {
    "metric": {
      "properties": {
        "jstream": {
          "type": "string"
        },
        "source": {
          "type": "string"
        },
        "source_attribute": {
          "type": "string"
        },
        "metric_name": {
          "type": "string"
        }
      }
    }
  }
}
```

```

    "metric_value": {
      "type": "long"
    },
    "metric_unit": {
      "type": "string"
    },
    "node_id": {
      "type": "string"
    },
    "count": {
      "type": "long"
    },
    "min": {
      "type": "double"
    },
    "avg": {
      "type": "double"
    },
    "max": {
      "type": "double"
    },
    "start_time": {
      "type": "date",
      "format": "yyyy-MM-dd'T'HH:mm:ss.SSSZ"
    },
    "interval_time": {
      "type": "integer"
    },
    "dump": {
      "type": "text"
    }
  }
}
}
}

```

Field	Description
tenant_id	A unique identifier the tenant.
jstream	Name of the doc type.
Source	Level at which the metric is getting captured. That is, ingestion, indexer, DAS, system, and so on.
source_attribute	An attribute of source. That is, source_attribute is topic name for source (kafka) and metric_name (kafka_topic_lag)
metric_name	is a metric name that Jarvis is capturing for specified component i.e. Ingestion_doc_count, cpu
metric_value	is a aggregated metric value captured over the period of specified interval time.
metric_unit	is a unit of metric value. i.e 'value', 'GB', 'percentage'
node_id	is a node information (kafka/ES node IP address)
count	is a total number of requests
avg	

Field	Description
	is a average of metric value for given interval, This field is not applicable for all metric.
min	is a min metric value over the period of interval, This field is not applicable for all metric.
max	is a max value over the period of interval, This field is not applicable for all metric.
interval_time	is pre defined interval in seconds, to accumulate the metric data before pushing it to the bus pipeline. Default value would be 60 seconds if not provided.
start_time	is a time when metric capturing started
dump	is a field where you can capture bulk information (i.e. comments, logs etc..)

DAS Queries

Use DAS queries to view the metrics collected using Jmetric. URL:

http://jarvis.com/das/jarvis/system/jmetrics_1.0

The following table lists the sample URLs to view metrics:

To Get	Use
Number of products onboarded.	/das/jarvis/system/jmetrics_1.0/?_filter=metric_name eq 'products_onboarded'
Number of tenants onboarded for product	das/CA_World/demo/jmetrics_1.0/?_filter=metric_name eq 'tenants_onboarded'
Number of doctypes onboarded for product	das/CA_World/_all/jmetrics_1.0/?_filter=metric_name eq 'doctypes_onboarded'
Number of documents ingested	das/CA_World/demo/jmetrics_1.0/?_filter=metric_name eq 'doc_ingested'
Documents that are ingested across tenants	das/CA_World/_all/jmetrics_1.0/?_filter=metric_name eq 'doc_ingested'
Documents that are ingested across products	/das/_all/_all/jmetrics_1.0/?_filter=metric_name eq 'doc_ingested'
Number of DAS query executed	/das/CA_World/demo/jmetrics_1.0/?_filter=metric_name eq 'das_query_count'
Response time for the DAS query	/das/CA_World/demo/jmetrics_1.0/?_filter=metric_name eq 'das_response_time'
Average response time for the DAS response in ms	/das/CA_World/demo/jmetrics_1.0/avg(field='metric_value')?_filter=metric_name eq 'das_response_time'
ElasticSearch metrics	/das/jarvis/system/jmetrics_1.0/?_filter=source eq 'elasticsearch'
ElasticSearch memory usage	/das/jarvis/system/jmetrics_1.0/?_filter=metric_name eq 'memory_usage'
ElasticSearch CPU usage	/das/jarvis/system/jmetrics_1.0/?_filter=metric_name eq 'cpu_usage'
ElasticSearch Shard count	

To Get	Use
	/das/jarvis/system/jmetrics_1.0/?_filter=metric_name eq 'shard_count'
ElasticSearch Index count	/das/jarvis/system/jmetrics_1.0/?_filter=metric_name eq 'index_count'
Number of documents that are indexed	/das/ca_world/demo/jmetrics_1.0?_filter=metric_name eq 'doc_indexed'
Number of documents that are failed at the indexer level	/das/ca_world/demo/jmetrics_1.0?_filter=metric_name eq 'doc_indexed_failed'
Number of payloads that are ingested	/das/ca_world/_all/jmetrics_1.0?_filter=metric_name eq 'payload_ingested'
Priority of payloads that are ingested	/das/ca_world/_all/jmetrics_1.0?_filter=metric_name eq 'payload_priority'
Size of Payloads that are ingested	/das/ca_world/_all/jmetrics_1.0?_filter=metric_name eq 'payload_size'
Number of ingested documents that are failed	/das/ca_world/demo/jmetrics_1.0?_filter=metric_name eq 'documents_failed_ingestion'
Number of documents that are ingested	/das/ca_world/demo/jmetrics_1.0?_filter=metric_name eq 'doc_ingested'
Size of the documents that are ingested	/das/ca_world/demo/jmetrics_1.0?_filter=metric_name eq 'docsize_ingested'

Back Up and Restore Elasticsearch Data

It is very important to back up the accumulated data as your cluster and your indexes grow. The Snapshot and Restore module of Elasticsearch allows you to create snapshots (backups) of individual indices or an entire cluster into a file system or into a remote repository like Amazon Simple Storage Service (Amazon S3). Using snapshots, you can create incremental backups, you can restore complete backups or just a few indices. The es-utils snapshot service of CA Jarvis schedules a job to take backup of indices in the Elasticsearch cluster. This service replaces the old snapshot with the new one every time it runs. Restoring of the index should be done using ES restore service. For more information on Snapshot and Restore, see [Elasticsearch documentation](#).

Select one of the following methods as per your requirement for back up and restore:

- [Create Snapshot and Back Up It to the S3 Registry](#)
- [Back Up Elasticsearch Data Using File System](#)
- [Restore Elasticsearch Data](#)

Create Snapshot and Back Up It to the S3 Registry

Follow these steps:

1. Change to the directory where you have unzipped the Jarvis installer binaries during installation, and complete the following sub-steps to edit the utils.properties file:

- a. Run the following command, and press **i** (lowercase I) in the keyboard to change to the insert mode:


```
vi utils.properties
```
 - b. Make the following changes in the `utils.properties` file:
 - i. **access_key**: AWS access key id required for authentication
 - ii. **secret_key**: AWS secret key required for authentication
 - iii. **bucket**: Name of the AWS S3 bucket to be used for snapshots
 - iv. **region**: Region where the bucket is located. Defaults to US Standard
 - v. **type**: s3
 - c. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes made to the `utils.properties` file.
2. Go to services and click S3 to see the snapshot in aws console.
You can see your bucket, the snapshot is available inside the bucket.
 3. Customize and run the following command to get the list of snapshot indexes by running:

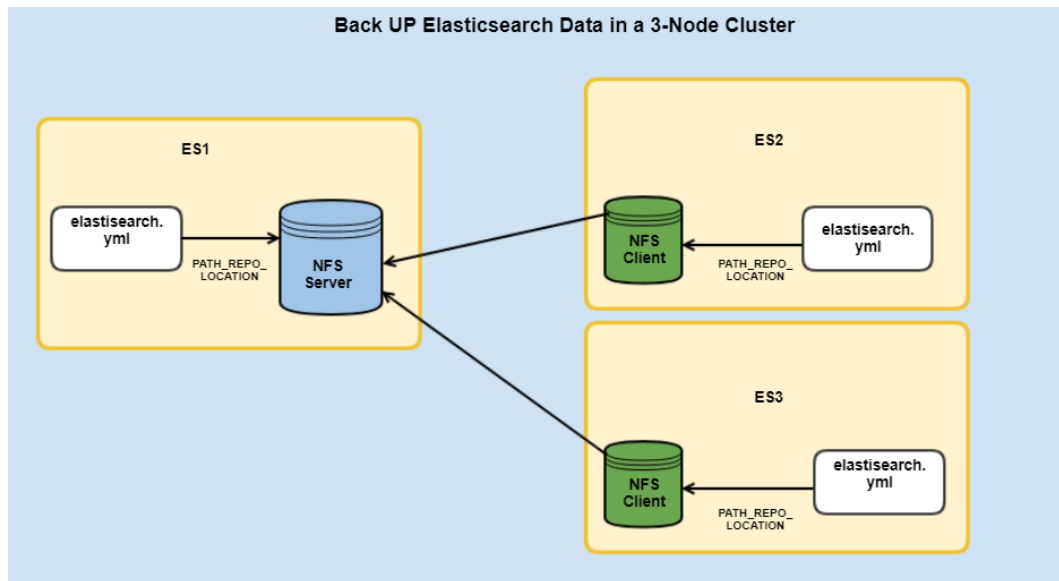

```
curl -XGET "http://localhost:9200/_snapshot/<Repository name>/_all"?pretty
```

Back Up Elasticsearch Data Using File System

- [Back Up Elasticsearch Data \(Docker Swarm Multiple Nodes\)](#)
 - [NFS Server-Side \(ES1\) Configurations](#)
 - [NFS Client-Side \(ES2 and ES3\) Configurations](#)
 - [YML Changes](#)
 - [Verify Repository Creation](#)

Back Up Elasticsearch Data (Docker Swarm Multiple Nodes)

To back up an Elasticsearch cluster data, a shared location is required and the shared location must be mounted on all the Elasticsearch nodes that are part of the cluster. We use **backup** as the name of the shared location throughout this procedure. The following diagram shows a 3-node cluster with ES1, ES2, and ES3 as the nodes.



- [Back Up Elasticsearch Data \(Docker Swarm Multiple Nodes\)](#)
 - [NFS Server-Side \(ES1\) Configurations](#)
 - [NFS Client-Side \(ES2 and ES3\) Configurations](#)
 - [YML Changes](#)
 - [Verify Repository Creation](#)

NFS Server-Side (ES1) Configurations



The IP addresses of the client nodes is required to complete this procedure. So, ensure to note down the IP addresses.

1. Run the following command to install NFS packages:

```
yum install nfs-utils libnfsidmap
```

2. Run the following commands one-by-one to enable and start NFS services:

```
systemctl enable rpcbind
```

```
systemctl enable nfs-server
```

```
systemctl start rpcbind
```

```
systemctl start nfs-server
```

```
systemctl start rpc-statd
```

```
systemctl start nfs-idmapd
```

3. Run the following command to create a directory named *backup* inside */opt/ca/Elasticsearch*. You will be sharing this directory with the client servers (ES2 and ES3).

```
mkdir -p /opt/ca/Elasticsearch/backup
```

4. Run the following command to give client servers read and write permissions on the created directory:

```
chmod 777 -R /opt/ca/Elasticsearch/backup
```

5. Modify “**/etc/exports**” file to include the directory that you want to share. Do the following steps:

- a. Run the following command, and press i (lowercase I) in the keyboard to change to the insert mode:

```
vi /etc/exports
```

- b. Add the following line for each client nodes. In our example, we will be adding two lines (for ES2 and ES3) :

```
/backup <ip address of the client>(rw,sync,no_root_squash)
```

Example:

For ES2:

```
/backup 192.0.2.1(rw,sync,no_root_squash)
```

For ES3:

```
/backup 192.0.3.3(rw,sync,no_root_squash)
```

- c. Press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes that are made to the **/etc/exports** file.

6. Run the following command to export the shared directories:

```
[root@server ~]# exportfs -r
```

7. Run the following commands one-by-one to configure firewall on the NFS server to allow client servers to access NFS shares:

```
firewall-cmd --permanent --zone public --add-service mountd
```

```
firewall-cmd --permanent --zone public --add-service rpc-bind
```

```
firewall-cmd --permanent --zone public --add-service nfs
```

```
firewall-cmd --reload
```

NFS Client-Side (ES2 and ES3) Configurations



The IP addresses of the server node is required to complete this procedure. So, ensure to note down the IP address.



Ensure that you repeat this procedure on all of your client nodes.

1. Run the following command to install NFS packages:

```
yum -y install nfs-utils libnfsidmap
```

2. Run the following commands one-by-one to enable and start NFS services:

```
systemctl enable rpcbind
```

```
systemctl start rpcbind
```

3. Run the following command to check the available shares on the NFS server before you start mounting:

```
showmount -e <ip address of the server>
```

Sample Output:

```
Export list for server:
/backup 192.168.12.5
```

4. Run the following command to create a directory named **backup** in the `opt/ca/Elasticsearch` path. Ensure that the shared directory in the NFS server and the directory that you are going to create have the same name:

```
mkdir opt/ca/Elasticsearch/backup
```

5. Run the following command to mount the shared directory **backup** from the NFS server in the **opt/ca/Elasticsearch/backup** directory of the NFS Client:

```
mount <ip address of the server>:/<name of the shared directory> /path
/<name of the shared directory>
```

Example:

```
mount 192.168.12.5:/backup /opt/ca/Elasticsearch/backup
```

6. Run the following command to verify the mounted share on client:

```
mount | grep nfs
```

7. Do the following steps to mount the shares automatically on every reboot:

- a. Run the following command, and press `i` (lowercase `I`) in the keyboard to change to the insert mode:

```
vi /etc/fstab
```

- b. Customize and Insert the following line at the end of the file:

```
<ip address of the server>:/<name of the shared directory> /path
/<name of the shared directory> nfs rw, sync, hard, intr 0 0
```

Example:

```
192.168.12.5:/backup/ /opt/ca/Elasticsearch/backup nfs rw, sync, hard,
intr 0 0
```

- c. Press the **Esc** key in the keyboard, type `:wq`, and press **Enter** to save the changes that are made to the `/etc/fstab` file.

YML Changes

Add the following parameter to the Elasticsearch service of your installation yml file:

```
PATH_REPO_LOCATION=/opt/ca/Elasticsearch/backup
```

Verify Repository Creation

Customize and use the following curl command in any one of the Elasticsearch nodes to verify the repository to store the snapshots :

```
curl -XPUT 'http://<ES_hostname>:<port>/_snapshot/my_backup?pretty'
{
  "my_backup" : {
```

```
"type" : "fs",
"settings" : {
  "compress" : "true",
  "location" : "/opt/ca/elasticsearch/backup"
}
}
```

Note:

my_backup is the repository name.

If the command returns true, then the repository registration is successful.

The es-utils snapshot service of CA Jarvis schedules a job to take backup of indices in the Elasticsearch cluster. This service replaces the old snapshot with the new one every time it runs (incremental backup).

Once this setup is done, by default, the snapshot job runs everyday at 23:00. You can update the following parameter in the *Utils* service (for Dockerswarm) or *esutils* (for Docker Single node) of your installation yml file to change the time when the snapshot triggers:

```
SNAPSHOT_CRON=0 0 23 * * ?
```

Restore Elasticsearch Data

- [Restore Elasticsearch Data on an Existing Cluster](#)
- [Restore Elasticsearch Data on a New Cluster](#)

Restore Elasticsearch Data on an Existing Cluster

1. Customize and run the following command in any one of the Elasticsearch nodes to list all the snapshots of the indexes:

```
curl -XPOST " <ES_hostname>:<port>/_snapshot/<Repository Name>/_all"
```

This command lists all the snapshots taken. In the following example, **two snapshots** are listed.

```
{
  "snapshots" : [
    {
      "snapshot" : "FB_virginia_loc1_account_plan_1.1_2017-08-27_23:00:11",
      "uuid" : "snFJQ_yeR4e-MHr_jxNfSw",
      "version_id" : 5020299,
      "version" : "5.2.2",
      "indices" : [
        "FB_virginia_cust1_account_plan_1.1"
      ],
      "state" : "SUCCESS",
      "start_time" : "2017-08-27T23:00:41.330Z",
      "start_time_in_millis" : 1503874841330,
      "end_time" : "2017-08-27T23:00:42.078Z",
      "end_time_in_millis" : 1503874842078,
      "duration_in_millis" : 748,
      "failures" : [ ],
      "shards" : {
        "total" : 5,
        "failed" : 0,
        "successful" : 5
      }
    },
    {
      "snapshot" : "FB_virginia_loc2_api_metrics_1.1_2017-08-27_23:00:11",
      "uuid" : "K1AZ-6LsRC6KpY407vr_9A",
      "version_id" : 5020299,
      "version" : "5.2.2",
      "indices" : [
```

```
"FB_virginia_loc2_api_metrics_1.1"
],
"state" : "PARTIAL",
"start_time" : "2017-08-27T23:01:02.707Z",
"start_time_in_millis" : 1503874862707,
"end_time" : "2017-08-27T23:01:07.531Z",
"end_time_in_millis" : 1503874867531,
"duration_in_millis" : 4824,
"failures" : [
  "shards" : {
    "total" : 5,
    "failed" : 1,
    "successful" : 4
  }
],
}
```

2. Customize and run the following command to restore individual snapshots of the indexes :

```
curl -XPOST " <ES_hostname>:<port>/_snapshot/<Repository Name>/<Snapshot Name>/_restore"
Example: To restore the first snapshot listed in the previous example, run the following command:
curl -XPOST "127.0.0.1:9200/_snapshot/my_backup/FB_virginia_loc1_account_plan_1.1_2017-08-27_23:00:11/_restore"
```

"acknowledged: true" indicates that the restore has been completed successfully.

3. Repeat the previous command for the rest of the snapshots that you want to restore.

Restore Elasticsearch Data on a New Cluster

To restore a snapshot on a new Elasticsearch cluster, a shared file-system that is accessible by all the nodes in the cluster is required. Also, ensure that the Repository Name should be the same. For example, if you take snapshot with backup as repository, you need to register a repository with the same name on new cluster.

In the following steps, we assume that the shared file-system is mounted to /opt/ca/elasticsearch/backup

Follow these steps:

1.
 - a. Register a repository with the same name as the snapshot using the following command. This example assumes that the snapshot is created using *my_backup* as the repository name.

```
curl -XGET '<ES_hostname>:<port>/_snapshot/my_backup?pretty'
{
  "my_backup" : {
    "type" : "fs",
    "settings" : {
      "compress" : "true",
      "location" : "/opt/ca/elasticsearch/backup"
    }
  }
}
```

- b. (Optional) Run the following command to list all the nodes where repository was successfully verified.

```
curl -XPOST 'http://localhost:9200/_snapshot/my_backup/_verify?pretty'
where my_backup is the repository name.
```

The verification process fails when the shared location is not accessible to other nodes that are part of the cluster.

- c. Move the data from the cluster where the snapshot is taken to fresh cluster. Data should be moved from the backup location of the old cluster to the current backup location.
- d. Customize and run the following command to restore a snapshot.

```
curl -XPOST"localhost:9200/_snapshot/my_backup/<snapshot_1/_restore"
```

where my_backup is the repository name and snapshot_1 is the name of the snapshot.

Example:

```
curl -XPOST"localhost:9200/_snapshot/my_backup/jarvisindex_1.7_1_2017-03-14_05:10:55/_restore"
```

In this example, the Snapshot location will be /opt/ca/elasticsearch/backup. You will see meta-data and snapshot files inside the directory. Snapshot name will be similar as shown below:

Index name prepended with snap and suffixed with creation date and time.

Example: snap-jarvisindex_1.7_1_2017-03-14_05:10:55.dat

Using

- [API Reference](#)
- [CA Jarvis Dashboard](#)

API Reference

The following Jarvis RESTful Application Programming Interfaces (APIs) help the application developers integrate their products with Jarvis. These APIs help applications to feed data into Jarvis for deriving insights from the data, and to access both data and insights (analytics information) from Jarvis. This document describes each API and how you can use the API to leverage the capabilities of Jarvis.

- [Product Onboarding API](#)
- [Tenant Onboarding API](#)
- [Mapping API](#)
- [Data Ingestion API](#)
- [Data Access Service API \(DAS\)](#)
- [DocView API](#)
- [Speed Job Onboarding API](#)
- [Batch Job Onboarding API](#)
- [Health Check API](#)



Ensure that the applications that consume CA Jarvis APIs follow protection measures against cross-site scripting (XSS) attacks or other injection issues.



Refer the **analyticsInstaller.properties** file to know the <host-name> and <port number> to form the URLs while using the APIs.

Product Onboarding API

The Product Onboarding API helps you do the following for products that are registered with Jarvis:

- [Create a Product](#)
- [Update a Product](#)
- [Get All Products](#)
- [Get Specific Product](#)

Make the following changes based on your environment (SSL or non-SSL) :

SSL	Non-SSL
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.
curl -XGET 'https://localhost:9200/' --cacert <certificate of CA> as the command for querying Elasticsearch.	curl -XGET 'http://localhost:9200/' as the command for querying Elasticsearch.



Before doing onboarding and ingestion using **Postman client** in an SSL-enabled setup, see [this procedure](#).



If retention_period is set at multiple levels, the following precedence is followed while determining which value should be considered:

1. tenant-doc_type level
2. tenant level
3. doc_type level
4. product_level
5. purge default

Create a Product

Request:

URI	HTTP Method
/products	POST

This API creates a new document in the meta-data index with type="product_details" and include the timestamp of when it was created.


Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/products

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	marvel
product_name	Name of the product that you are planning to onboard.	Yes	String	Marvel

Name	Description	Required?	Type	Example
product_description	A brief description about the product.	Optional	String	Analytics Engine
product_details	Additional parameter to hold more information about the product .	Optional	Json	-
retention_period	The number of days the ingested document should be kept in Jarvis. This value will be used by the Purge service while deleting old documents. If you keep 0 as the retention_period, the Purge service will not delete any ingested data.	Optional	Integer	10

ldds	Additional details (doc_type_filter and google_map_key) of the product to be onboarded to CA Jarvis Dashboard.  If you do not provide these details (doc_type_filter and google_map_key), the product will be onboarded to CA Jarvis Dashboard with the default values. If you do not want to onboard the product to CA Jarvis Dashboard, remove the parameter (ldds).	Values are optional. If you do not want to onboard the product to CA Jarvis Dashboard, remove the parameter (ldds).	Json	{ "doc_type_filter": "<regex>", "google_map_key": "" }
------	---	---	------	---

If you want to onboard the product to CA Jarvis Dashboard later, use the PATCH request of the CA Jarvis product onboarding API.

docTypeFilter :

Provide a regular expression to filter the DocTypes. All DocTypes whose name matches the regular expression provided are excluded from CA Jarvis Dashboard operations. For example, if you specify `apim_*`, all the jstreams starting with `apim` are excluded. This parameter filters only the viewlet and filter authoring system. If an imported dashboard references an Index that is specified to be filtered out, the dashboard displays the data.

googleMapKey:

Google Maps require an API key to make it functional. Provide the key, if available. If not, leave it empty.

Sample Request

POST /products

```
{
  "product_id": "marvel",
  "product_name": "Marvel",
  "product_description": "Analytics Engine", "retention_period": 10
  "ldds":{"doc_type_filter": "<regex>", "google_map_key": ""}
}
```

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  "_message": "product id already exists"
  "_code": 1001
}
```

Update a Product

Request:

URI	HTTP Method
/products	PATCH


This API updates an existing product in the meta-data index.

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/products

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	marvel
product_name	New name of the product tto be updated.	Optional	String	Marvel
product_description	Updated description about the product.	Optional	String	Analytics Engine
product_details	Updated parameter to hold more information about the product .	Optional	Sting	-
retention_period		Optional	Integer	10

Name	Description	Required?	Type	Example
	<p>The number of days the ingested document should be kept in Jarvis. This value will be used by the Purge service while deleting old documents.</p> <p>If you keep 0 as the retention_period, the Purge service will not delete any ingested data.</p>			
ldds	<p>Additional details (doc_type_filter and google_map_key) of the product to be onboarded to CA Jarvis Dashboard.</p> <div data-bbox="528 663 967 1079" data-label="Complex-Block"> <p> If you do not provide these details (doc_type_filter and google_map_key), the product will be onboarded to CA Jarvis Dashboard with the default values.</p> <p>If you do not want to onboard the product to CA Jarvis Dashboard, remove the parameter (ldds).</p> </div>	Values are optional. If you do not want to onboard the product to CA Jarvis Dashboard, remove the parameter (ldds).	Json	<pre>{ "doc_type_filter": "<regex>", "google_map_key": "" }</pre>
	<p>docTypeFilter : Provide a regular expression to filter the DocTypes. All DocTypes whose name matches the regular expression provided are excluded from CA Jarvis Dashboard operations. For example, if you specify apim_*, all the jstreams starting with apim are excluded. This parameter filters only the viewlet and filter authoring system. If an imported dashboard references an Index that is specified to be filtered out, the dashboard displays the data.</p> <p>googleMapKey: Google Maps require an API key to make it functional. Provide the key, if available. If not, leave it empty.</p>			

Sample Request

PATCH /products

```
{
  "product_id": "marvel",
  "product_name": "Marvel",
  "product_description": "Analytics Engine", "retention_period": 10
  "ldds":{"doc_type_filter": "<regex>", "google_map_key": ""}
}
```

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  "_message": "product id already exists"
  "_code": 1001
}
```

Get All Products

Request:

URI	HTTP Method
/products	GET

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/products

Sample Request

```
GET /products
```

Response:

Success	200
Error	500

Sample Response

```
{
  "products": [
    {
      "product_id": "marvel",
      "product_description": "Analytics Engine",
      "product_details": null,
      "product_name": "Marvel"
    }
  ]
}
```

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  "_message": "product id invalid"
  "_code": 1001
}
```

Get Specific Product

Request:

URI	HTTP Method
/products(<product_id>)	GET

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/products
(product_id=<product_id>)

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	marvel

Sample Request

```
GET /products(product_id='marvel')
```

Response:

Success	200
Error	400

Sample Response

```
{
  "product_id": "marvel",
  "product_name": "Marvel",
  "product_description": "Analytics Engine",
  "product_details": null
}
```

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  "_message": "product id invalid"
  "_ code": 1001
}
```

Tenant Onboarding API

The Tenant Onboarding API helps you do the following:

- [Create a Tenant](#)
- [Update a Tenant](#)
- [Get All Tenants of a Product](#)
- [Get Specific Tenant Details](#)
- [Delete a Tenant](#)

Make the following changes based on your environment (SSL or non-SSL) :

SSL	Non-SSL
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.
curl -XGET 'https://localhost:9200/' --cacert <certificate of CA> as the command for querying Elasticsearch.	curl -XGET 'http://localhost:9200/' as the command for querying Elasticsearch.



Before doing onboarding and ingestion using **Postman client** in an SSL-enabled setup, see [Troubleshooting](#).



If retention_period is set at multiple levels, the following precedence is followed while determining which value should be considered:

1. tenant-doc_type level
2. tenant level
3. doc_type level
4. product_level
5. purge default

Create a Tenant


Request:

URI	HTTP Method
/onboarding/tenants	POST

Resource URL

http(s)://<host name or IP address>:<port number>/onboarding/tenants

Parameters:

Name	Description	Required?	Type	Example
product_id	ID of an existing product	Yes	String	marvel
tenant_id	A unique identifier for the tenant.	Yes	String	xmen
tenant_details	A brief description about the tenant. You can provide the number of days a the retention period.	Optional	JSON	
retention_period	The number of days the ingested document should be kept in Jarvis. This value will be used by the Purge service while deleting old documents. If you keep 0 as the retention_period, the Purge service will not delete any ingested data.	Optional	Integer	10
tenant_doc_type_details	Retention period per doc type for the current tenant.	Optional	JSON Array	[{"doc_type_id": "events", "doc_type_version": 1", "retention_period": 5 }]
ldds	Additional details (product_admin_role_name, admin_role_name, user_role_name, and ana) of the tenant to be onboarded to CA Jarvis Dashboard.  If you do not want to onboard the tenant to CA Jarvis Dashboard, remove the parameter (ldds).	Optional. If you do not want to onboard the tenant to CA Jarvis Dashboard, remove the parameter (ldds).	json	{ "product_admin_role_name": "PA", "admin_role_name": "TA", "user_role_name": "OA", "ana": "internal" }

Name	Description	Required?	Type	Example
	<p>If you want to onboard the tenant to CA Jarvis Dashboard later, use the PATCH request of the CA Jarvis tenant onboarding API.</p> <p>product_admin_role_name</p> <p>Name for the user group of product admins.</p> <p>admin_role_name:</p> <p>Name for the user group of tenant admins.</p> <p>user_role_name:</p> <p>Name for the user group of normal users.</p> <p>ana: ana stands for Authentication and Authorization, This variable can have the following values values: Internal: To use CA Jarvis Dashboard as the identity store. Ingress: To use APIM as the identity store</p>			

Sample Request

POST /tenants

```
{
  "product_id": "marvel",
  "tenant_id": "xmen",
  "tenant_details": {
    "file_limit": 1024
  },
  "retention_period": 10,
  "tenant_doc_type_details": [
    {
      "doc_type_id": "events",
      "doc_type_version": "1",
      "retention_period": 5
    }
  ],
  "ldds": {
    "product_admin_role_name": "PA",
    "admin_role_name": "TA",
    "user_role_name": "OA",
  }
}
```



```
"ana": "internal"}

}
```

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  "_message": "invalid tenant_id "
  "_code": 1001
}
```

Update a Tenant

Request:


URI	HTTP Method
/onboarding/tenants	PATCH

Resource URL

http(s)://<host name or IP address>:<port number>/onboarding/tenants

Parameters:

Name	Description	Required?	Type	Example
product_id	ID of an existing product	Yes	String	marvel
tenant_id	A unique identifier for the tenant.	Yes	String	xmen
tenant_details	A brief description about the tenant. You can provide the number of days a the retention period.	Optional	JSON	
retention_period	The number of days the ingested document should be kept in Jarvis. This value will be used by the Purge service while deleting old documents.	Optional	Integer	10

Name	Description	Required?	Type	Example
	If you keep 0 as the retention_period, the Purge service will not delete any ingested data.			
tenant_doc_type_details	Retention period per doc type for the current tenant.	Optional	JSON Array	[{ "doc_type_id": "events", "doc_type_version": "1", "retention_period": 5 }]
ldds	Additional details (product_admin_role_name, admin_role_name, user_role_name, and ana) of the tenant to be onboarded to CA Jarvis Dashboard. <div>  If you do not want to onboard the tenant to CA Jarvis Dashboard, remove the parameter (ldds). </div>	Optional. If you do not want to onboard the tenant to CA Jarvis Dashboard, remove the parameter (ldds).	json	{ "product_admin_role_name": "PA", "admin_role_name": "TA", "user_role_name": "OA", "ana": "internal" }
product_admin_role_name Name for the user group of product admins.				
admin_role_name: Name for the user group of tenant admins.				
user_role_name: Name for the user group of normal users.				
ana: ana stands for Authentication and Authorization, This variable can have the following values values: Internal: To use CA Jarvis				

Name	Description	Required?	Type	Example
	Dashboard as the identity store. Ingress: To use APIM as the identity store			

Sample Request

PATCH/tenants

```
{
  "product_id": "marvel",
  "tenant_id": "xmen",
  "tenant_details": {
    "file_limit": 1024
  },
  "retention_period": 20,
  "tenant_doc_type_details": [
    {
      "doc_type_id": "events",
      "doc_type_version": "1",
      "retention_period": 5
    }
  ],
  "ldds": {
    "product_admin_role_name": "PA",
    "admin_role_name": "TA",
    "user_role_name": "OA",
    "ana": "internal"
  }
}
```

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  "_message": "invalid tenant_id "
  "_code": 1001
}
```

Get All Tenants of a Product

Request:

URI	HTTP Method
onboarding/tenants(product_id=<product_id>)	GET

Resource URL:

http(s)://<ip address>:<port number>/onboarding/tenants(product_id=<product_id>)

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	marvel

Sample Request

```
GET /tenants(product_id='marvel')
```

Response:

Success	200
Error	400

Sample Response

```
{
  "tenants": [
    {
      "tenant_id": "xmen",
      "product_id": "marvel",
      "tenant_details": {
        "file_limit": 1024
      }
    },
    {
      "tenant_id": "xmen2",
      "product_id": "marvel",
      "tenant_details": {
        "file_limit": 1024
      }
    }
  ]
}
```

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  "_message": "Product not found, provided product_id doesn't exist."
  "_code": "1002"
}
```

}

Get Specific Tenant Details

Request:

URI	HTTP Method
onboarding/tenants(product_id=<product_id>, tenant_id=<tenant_id>)	GET

Resource URL:

http(s)://<ip>:<port number>/onboarding/tenants(product_id=<product_id>, tenant_id=<tenant_id>)

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	marvel
tenant_id	A unique identifier for the tenant.	Yes	String	xmen

Sample Request

```
GET /tenants(product_id='marvel', tenant_id='xmen')
```

Response:

Success	200
Error	400

Sample Response

```
{
  "tenant_id": "xmen",
  "product_id": "marvel",
  "tenant_details": {
    "file_limit": 1024
  }
}
```

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  "_message": "Invalid product id "
  "_code": "1004"
}
```

Delete a Tenant

Request:

URI	HTTP Method
/onboarding/tenants	DELETE

Resource URL

http(s)://<host name or IP address>:<port number>/onboarding/tenants

Parameters:

Name	Description	Required?	Type	Example
product_id	ID of an existing product	Yes	String	marvel
tenant_id	A unique identifier for the tenant.	Yes	String	xmen

Sample Request

DELETE/tenants

```
{
  "product_id": "marvel",
  "tenant_id": "xmen"
}
```

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  "_message": "invalid tenant_id "
  "_code": "1004"
}
```

Mapping API

The Mapping API helps create structure of incoming data for a particular doc_type_id and doc_type_version for each product. Mapping is the process of defining how a document and its fields are stored and indexed. For example, use mappings to define which fields contain numbers, dates, or geolocations.

The mapping API helps you do the following:

- **Create Document Type Definition**
 - Request:
 - Resource URL:
 - Parameters:
- **Get All Document Type Definitions**
 - Request:
 - Resource URL:
 - Parameters:
 - Response:
 - Error Response:
- **Get Specific Document Type Definition**
 - Request:
 - Resource URL:
 - Parameters:
 - Response:
 - Error Response:
- **Update Document Type**
 - Request:
 - Error Response:

Jarvis schema definition is very similar to the schema definition of Elasticsearch. Here is a sample schema:

```
{
  "mappings": {
    "user": {
      "properties": {
        "name": {
          "properties": {
            "first": {
              "type": "string",
              "isArray": true
            }
          }
        },
        "user_id": {
          "type": "string",
          "index": "not_analyzed"
        }
      }
    }
  }
}
```



When you want to ingest an array instead of a single field, for example, ["one", "two"] or [1, 2], ensure that "isArray" is set as true. The ingested data for the fields with "isArray" attribute should be an array only and not a single field.

Mapping Types

Mapping types help divide the documents in an index into logical groups. Each mapping type has Meta-fields and Fields or properties. Meta-fields help you customize how the metadata of a document should be treated. Fields or properties are the list of fields or properties that are relevant to a mapping type. Each Field has its own datatype, which can be one of the following:

- Basic datatype like text , Boolean , date, and so on.
- A datatype that supports the hierarchical nature of JSON like, object or nested.
- A specialized datatype like geo_point.

Allowed Data-Types:

Data Type	Description	Example
text	Defines a field to index full-text and unstructured values such as the content of a wiki page or the description of a product. Before indexing, the fields that are defined as text are passed through an analyzer to convert the string into a list of individual terms.	Jarvis is an analytics engine.



Do not use this datatype, if you want to index structured content such as email address, status, zip code, Fully Qualified Domain Name, and so on . Use keyword as the datatype to index structured content.

The text fields accept the following parameters:

analyzer

Indicates the analyzer that should be used for analyzed text fields, both at index-time and at the search-time (unless overridden by the search_analyzer). Defaults to the default index analyzer, or the standard analyzer.

boost


Indicates the mapping field-level query time boosting. Accepts a floating point number. Defaults to 1.0.

eager_global_ordinals


Indicates whether the global ordinals should be loaded eagerly on refresh or not. Accepts true or false. Defaults to false. Enabling this is a good idea on fields that are frequently used for (significant) terms aggregations.

fielddata

Data Type	Description	Example
	<p>Indicates whether the field can use in-memory fielddata for sorting, aggregations, or scripting or not. Accepts true or false. Defaults to false.</p> <p>fielddata_frequency_filter</p> <p>Indicates the expert settings that allow to decide which values to load in memory when fielddata is enabled. By default, all values are loaded.</p> <p>fields</p> <p>Indicates the multi-fields that allow the same string value to be indexed in multiple ways for different purposes, such as one field for search and a multi-field for sorting and aggregations, or the same string value analyzed by different analyzers.</p> <p>include_in_all</p> <p>Indicates whether the field value should be included in the _all field or not. Accepts true or false. Defaults to false if index is set to no, or if a parent object field sets include_in_all to false. Otherwise, defaults to true.</p> <p>index</p> <p>Indicates whether the field be searchable or not. Accepts true or false. Defaults to true.</p> <p>index_options</p> <p>Indicates what information should be stored in the index, for search and highlighting purposes. Defaults to positions.</p> <p>norms</p> <p>Indicates whether the field-length should be taken into account when scoring queries or not. Accepts true or false. Defaults to true.</p> <p>position_increment_gap</p> <p>Indicates the number of fake term position that should be inserted between each element of an array of strings. Defaults to the position_increment_gap configured on the analyzer which defaults to 100. 100 was chosen because it prevents phrase queries with reasonably large slops (less than 100) from matching terms across field values.</p> <p>store</p>	

Data Type	Description	Example
	<p>Indicates whether the field value should be stored and retrievable separately from the <code>_source</code> field. Accepts true or false. Defaults to false.</p> <p>search_analyzer</p> <p>Indicates the analyzer that should be used at search time on analyzed fields. Defaults to the analyzer setting.</p> <p>search_quote_analyzer</p> <p>Indicates the analyzer that should be used at search time when a phrase is encountered. Defaults to the <code>search_analyzer</code> setting.</p> <p>similarity</p> <p>Indicates which scoring algorithm or similarity should be used. Defaults to BM25.</p> <p>term_vector</p> <p>Indicates whether term vectors should be stored for an analyzed field or not. Accepts true or false. Defaults to no.</p>	
keyword	<p>Defines a field to index structured values such as an email address, status, zip code, Fully Qualified Domain Name, and so on. Keyword fields are only searchable by their exact value. They are typically used for the following purposes:</p> <ul style="list-style-type: none"> ▪ Filtering. For example, to find all the emails that are in <i>Draft</i> status. ▪ Sorting. For example, to sort all the emails that came from support.ca.com. ▪ Aggregations. For example, to count the unique authors that match a query while indexing books. <div style="border: 1px solid #f0e68c; padding: 10px; margin-top: 10px;"> <p> Do not use this datatype, if you want to index unstructured content such as email content, product description, and so on. Use text as the datatype to index unstructured content.</p> </div> <p>The keyword fields accept the following parameters:</p> <p>boost</p> <p>Indicates the mapping field-level query time boosting. Accepts a floating point number, defaults to 1.0.</p>	www.ca.com

Data Type	Description	Example
	<p>doc_values</p> <p>Indicates whether the field should be stored on disk in a column-stride fashion or not. Storing the field on a disk in a column-stride fashion enables you to use the field later for sorting, aggregations, or scripting. Accepts true or false. Defaults to true.</p> <p>eager_global_ordinals</p> <p>Indicates whether the global ordinals should be loaded eagerly on refresh or not. Accepts true or false. Defaults to false. Enabling this is a good idea on fields that are frequently used for (significant) terms aggregations.</p> <p>fields</p> <p>Indicates the multi-fields that allow the same string value to be indexed in multiple ways for different purposes, such as one field for search and a multi-field for sorting and aggregations, or the same string value analyzed by different analyzers.</p> <p>ignore_above</p> <p>Do not index any string longer than this value. Defaults to 2147483647 so that all values would be accepted.</p> <p>include_in_all</p> <p>Indicates whether the field value should be included in the _all field or not. Accepts true or false. Defaults to false if index is set to no, or if a parent object field sets include_in_all to false. Otherwise, defaults to true.</p> <p>index</p> <p>Indicates whether the field be searchable or not. Accepts true or false. Defaults to true.</p> <p>index_options</p> <p>Indicates what information should be stored in the index, for scoring purposes. Defaults to docs but can also be set to freqs to take term frequency into account when computing scores.</p> <p>norms</p> <p>Indicates whether the field-length should be taken into account when scoring queries or not. Accepts true or false. Defaults to false.</p> <p>null_value</p>	

Data Type	Description	Example
	<p>Indicates explicit null values. Accepts a string value which is substituted for any explicit null values. Defaults to null, which means the field is treated as missing.</p> <p>store</p> <p>Indicates whether the field value should be stored and retrievable separately from the <code>_source</code> field or not. Accepts true or false. Defaults to false.</p> <p>similarity</p> <p>Indicates which scoring algorithm or similarity should be used. Defaults to BM25.</p>	
<div>  Note: For byte, short, integer, and long types, pick the smallest type which is enough for your use-case. This ensures more efficient indexing and searching. However that given that storage is optimized based on the actual values that are stored, picking one type over another one will have no impact on storage requirements. </div>		
byte	A signed 8-bit integer with a minimum value of -128 and a maximum value of 127.	
short	A signed 16-bit integer with a minimum value of -32,768 and a maximum value of 32,767.	
integer	A signed 32-bit integer with a minimum value of -2 ³¹ and a maximum value of 2 ³¹ -1.	
long	A signed 64-bit integer with a minimum value of -2 ⁶³ and a maximum value of 2 ⁶³ -1.	
float	<p>A single-precision 32-bit IEEE 754 floating point. Approximate range: ±1.5e-45 to ±3.4e38.</p> <p>Datatypes float and double consider that -0.0 and +0.0 are different values. So, doing a term query on -0.0 will not match +0.0 and vice-versa. Same is true for range queries. If the upper bound is -0.0 then +0.0 will not match, and if the lower bound is +0.0 then -0.0 will not match.</p>	
double	<p>A double-precision 64-bit IEEE 754 floating point. Approximate range: ±5.0e-324 to ±1.7e308.</p> <p>Datatypes float and double consider that -0.0 and +0.0 are different values. So, doing a term query on -0.0 will not match +0.0 and vice-versa. Same is true for range queries. If the upper bound is -0.0 then +0.0 will not match, and if the lower bound is +0.0 then -0.0 will not match.</p>	

Data Type	Description	Example
	Double has 15 -16 decimal digits of precision, while float has 7. So, the inaccuracies will be smaller in the double datatype.	
ip	<p>An ip field can index or store either IPv4 or IPv6 addresses. The most common way to query ip addresses is to use the Classless Inter-Domain Routing (CIDR) notation: <i>ip_address</i>[/<i>prefix_length</i>].</p> <p>Example: In a query, 192.168.100.14/24 represents the IPv4 address 192.168.100.14 and its associated routing prefix 192.168.100.0, or equivalently, its subnet mask 255.255.255.0, which has 24 leading 1-bits.</p> <p>The ip fields accept the following parameters:</p> <p>boost</p> <p>Indicates the mapping field-level query time boosting. Accepts a floating point number, defaults to 1.0.</p> <p>doc_values</p> <p>Indicates whether the field should be stored on disk in a column-stride fashion or not. Storing the field on a disk in a column-stride fashion enables you to use the field later for sorting, aggregations, or scripting. Accepts true or false. Defaults to true.</p> <p>include_in_all</p> <p>Indicates whether the field value should be included in the <i>_all</i> field or not. Accepts true or false. Defaults to false if index is set to no, or if a parent object field sets <i>include_in_all</i> to false. Otherwise, defaults to true.</p> <p>index</p> <p>Indicates whether the field be searchable or not. Accepts true or false. Defaults to true.</p> <p>null_value</p> <p>Indicates an IPv4 value that is substituted for any explicit null values. Defaults to null, which means the field is treated as missing.</p> <p>store</p> <p>Indicates whether the field value should be stored and retrievable separately from the <i>_source</i> field or not. Accepts true or false. Defaults to false.</p>	192.168.100.14
boolean	<p>Boolean fields accept the following values:</p> <ul style="list-style-type: none"> ▪ JSON true and false. 	

Data Type	Description	Example
	<ul style="list-style-type: none"> strings and numbers, which are interpreted as either true or false. <p>False Values:</p> <p>For indexing: false, "false", "off", "no", "0", "" (empty string), 0, or 0.0.</p> <p>For Searching: "false" or false.</p> <p>True Values:</p> <p>For indexing and searching: "true" or true.</p> <p>The boolean fields accept the following parameters:</p> <p>boost</p> <p>Indicates the mapping field-level query time boosting. Accepts a floating point number, defaults to 1.0.</p> <p>doc_values</p> <p>Indicates whether the field should be stored on disk in a column-stride fashion or not. Storing the field on a disk in a column-stride fashion enables you to use the field later for sorting, aggregations, or scripting. Accepts true or false. Defaults to true.</p> <p>index</p> <p>Indicates whether the field be searchable or not. Accepts true or false. Defaults to true.null_value</p> <p>Accepts any of the true or false values listed previously. The value is substituted for any explicit null values. Defaults to null, which means the field is treated as missing.</p> <p>store</p> <p>Indicates whether the field value should be stored and retrievable separately from the _source field or not. Accepts true or false. Defaults to false.</p>	
date	<p>JSON does not have a date datatype. So, date datatype in Elasticsearch can be in one of the following format:</p> <ul style="list-style-type: none"> <ul style="list-style-type: none"> yyyy-MM-dd'T'HH:mm:ss.SSSZ yyyy-MM-dd'T'HH:mm:ssZ yyyy-MM-dd'T'HH:mmZ yyyy-MM-dd epoch_millis 	

Data Type	Description	Example
Date	ter Date Component	Presentation Example
	Year	Year 1996
	Month in year	Month July; Jul; 07
	Day in month	Number 10
	Hour in day (0-23)	Number 0
	Minute in hour	Number 30
	Second in minute	Number 55
	Millisecond	Number 978
	Time zone	RFC 822 time zone -0800
	Time zone	ISO 8601 time zone -08; -0800; -08:00

If no other format is mentioned, epoch_millis is considered as the format. Internally, dates are converted to UTC (if the time-zone is specified) and stored as a long number representing milliseconds-since-the-epoch.

geo_point	<p>The geo_point fields accept latitude-longitude pairs. Latitude is a decimal number between -90.0 and +90.0. Longitude is a decimal number between -180.0 and +180.0. Following are the purposes of geo_point fields:</p> <ul style="list-style-type: none"> ▪ To find geo-points within an area defined by two longitudes and two latitudes (a bounding box), within a certain distance of a central point, or within a polygon. ▪ To aggregate documents by geographically or by distance from a central point. ▪ To integrate distance into the relevance score of a document. ▪ To sort documents by distance. 	<p>Jarvis validates "geo_point" to support the following formats:</p> <p>Geo-point as an object: A json object containing latitude and longitude keys-value pairs.</p> <p>Example:</p> <pre>"latde": 41.12, "lontd": -71.34</pre> <p>Geo-point as a string: Comma-separated latitude and longitude values.</p> <p>For Geo-point as a string, the order is latitude, longitude .</p> <p>Example:</p> <pre>"41.12,-71.34"</pre>
-----------	--	---

Data Type	Description	Example
		<p>Geo-point as an array: A json array containing longitude and latitude.</p> <p>For Geo-point as an array, the order is longitude, latitude.</p> <p>Example:</p> <p>[-71.34, 41.12]</p>
binary	<p>The binary type accepts a binary value as a Base64 encoded string. The field is not stored by default and is not searchable. This type is used for encoding binary data that needs be stored and transferred over media that are designed to deal with textual data.</p> <p>The Base64 encoded binary value must not have embedded newlines(\n).</p>	TWFuIGZlZGRpc3Rpbmd1aX
object	<p>Object data type represents JSON objects (inner objects) embedded inside your parent document.</p> <p>The object fields accept the following parameters:</p> <p>dynamic Indicates whether new properties should be added dynamically to an existing object or not. Accepts true, false, and strict. Defaults to true.</p> <p>enabled Indicates whether the JSON value given for the object field should be parsed and indexed (true) or completely ignored (false). Defaults to true.</p> <p>include_in_all Sets the default include_in_all value for all the properties within the object. The object itself is not added to the _all field.</p> <p>properties Indicates the fields within the object, which can be of any datatype, including object. New properties may be added to an existing object.</p>	<p>"name":"Tom", "car":{ "make":"Sedan", "model":"RS3"</p> <p>The car field is an object field with the inner object having two properties (make and model.)</p>
nested	<p>The nested type is a specialized version of the object datatype that allows arrays of JSON objects to be indexed and queried independently of each other. To index arrays of JSON objects and to maintain the independence of each object in the array, you should use the nested datatype instead of the object datatype. Internally, nested objects index each object in the array as a separate hidden document, meaning that each nested object can be queried independently of the others.</p> <p>Because nested documents are indexed as separate documents, they can only be accessed within the scope of the nested query, the nested/reverse_nested, or nested inner hits. Indexing a document with 100 nested</p>	<pre>{ "person":{ "properties":{ "name" : { "type" : "string" }, "car":{ "type" : "nested" } } } }</pre>

Data Type	Description	Example
	fields actually indexes 101 documents as each nested document is indexed as a separate document. To safeguard against ill-defined mappings the number of nested fields that can be defined per index has been limited to 50.	
	The nested fields accept the following parameters:	
	dynamic	
	Indicates whether new properties should be added dynamically to an existing nested object or not. Accepts true, false, and strict. Defaults to true.	
	include_in_all	
	Sets the default include_in_all value for all the properties within the nested object. Nested documents do not have their own _all field. Instead, values are added to the _all field of the main "root" document.	
	properties	
	Indicates the fields within the nested object, which can be of any datatype, including nested. New properties may be added to an existing nested object.	

Allowed Characters For Parameter Values

Parameter values in mapping cannot be empty (" "). Following are the allowed characters for parameter values:

- A-Z
- a-z
- 0-9 (Allowed anywhere other than in the beginning of the parameter name.)
- _ (Allowed anywhere other than in the beginning of the parameter name.)

Make the following changes based on your environment (SSL or non-SSL) :

SSL	Non-SSL
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.
curl -XGET 'https://localhost:9200/' --cacert <certificate of CA> as the command for querying Elasticsearch.	curl -XGET 'http://localhost:9200/' as the command for querying Elasticsearch.



If retention_period is set at multiple levels, the following precedence is followed while determining which value should be considered:

1. tenant-doc_type level
2. tenant level
3. doc_type level
4. product_level
5. purge default

Create Document Type Definition

Request:

URI	HTTP Method
/doc_type	POST

This API creates a new document in the meta-data index with type="product_details" and include the timestamp of when it was created.

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/doc_type

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	marvel
doc_type_id	A unique identifier for your document type.	Yes	String	tweet
doc_type_version	Version of the document type.	Yes	String	1.0
unique_id	A valid JSON Path.	No	String	['stream_id']
mappings	Information about the data fields that will be ingested.	Yes	JSON structure	<i>See sample request below</i>
retention_period	The number of days the ingested document should be kept in Jarvis. This value will be used by the Purge service while deleting old documents. If you keep 0 as the retention_period, the Purge service will not delete any ingested data.	No	Integer	10
tenant_doc_type_details	Retention period per tenant for the current doc type.	No	JSON Array	[{"tenant_id": "t2", "retention_period": 2}]
data_routing		No	Boolean	{

Name	Description	Required? Type	Example
	This parameter helps you specify the service to which the ingested data should be routed. The following table summarizes how data is routed based on the values that you set.		<pre>"service_layer" : true, "batch_layer" : true }</pre>
ue Set	Description		
vice_layer=true	Data will be routed to the		
ch_layer=false	Service Layer (Elasticsearch). This is the default value for the data_routing parameter.		
vice_layer=false	Data will be routed to the		
ch_layer=true	Batch Layer (HDFS).		
vice_layer=false	In this case, data is not		
ch_layer=false	routed to the Service Layer (Elasticsearch) or to the Batch Layer (HDFS). Use this configuration, if you want data to be routed and stored only in the Speed Layer (Kafka).		
vice_layer=true	Data will be routed to both		
ch_layer=true	the Service Layer (Elasticsearch) and the Batch Layer (HDFS).		

Note: The field marked as "unique_id" should be of type String only.

Sample Request

POST /doc_type

```
{
  "product_id": "DXA",
```

```

"doc_type_id": "WB",
"doc_type_version": "1.1",
"retention_period": 10,
"unique_id": "${'geolocation'}['timeStamp']",
"mappings": {
  "jstream": {
    "properties": {
      "firstname": {
        "type": "integer"
      },
      "lastname": {
        "type": "string",
        "index": "not_analyzed"
      },
      "about": {
        "type": "string",
        "index": "analyzed"
      },
      "geolocation": {
        "type": "object",
        "properties": {
          "timeStamp": {
            "type": "integer"
          },
          "geopoint": {
            "type": "geo_point"
          },
          "altitude": {
            "type": "integer"
          },
          "longitude": {
            "type": "integer"
          },
          "latitude": {
            "type": "integer"
          },
          "altitudeAccuracy": {
            "type": "integer"
          },
          "speed": {
            "type": "integer"
          },
          "heading": {
            "type": "integer"
          },
          "accuracy": {
            "type": "integer"
          },
          "created_at": {
            "type": "date",
            "format": "yyyy-MM-dd"
          }
        }
      }
    }
  }
}

```

```
"tenant_doc_type_details":[
  { "tenant_id": "t2", "retention_period" : 2}]
}
```

_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```
{
  " _message": "doc_type_id and doc_type_version combination already exists"
  " _code": 400
}
```

Get All Document Type Definitions

Request:

URL	HTTP Method
/doc_type(product_id=<product_id>)	GET

Directly queries the meta-data index with type="mapping_info" by matching product_id field. It also takes an optional parameter "_flatten", which flattens the mapping response till the leaf node (type).

The URL also takes an optional parameter "_flatten," which flattens the mapping response till the leaf node. This returns a flat response for mapping with nested properties. In case of geo-point, the response will be flattened into lat, lon. In case of "nested" type, the response will be flattened until the nested field and it will also be flattened underneath the nested field (see Complex response with nested flattening in Sample response with _flatten example in Get Specific Document Type Definition).

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/doc_type
(product_id=<product_id>)

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	marvel
_flatten	Flatten the response till type	No	Boolean	_flatten=true _flatten=false

Sample Request

```
GET /doc_type(product_id='marvel')
```

Response:

Success	200
Error	400

Sample Response

```

{
  "doc_types": [
    {
      "mappings": {
        "jstream": {
          "properties": {
            "sentiment": {
              "index": "not_analyzed",
              "type": "string"
            },
            "friends_count": {
              "index": "not_analyzed",
              "type": "integer"
            },
            "version": {
              "index": "not_analyzed",
              "type": "string"
            },
            "userid": {
              "index": "not_analyzed",
              "type": "string"
            },
            "twitterid": {
              "index": "not_analyzed",
              "type": "string"
            },
            "url": {
              "index": "not_analyzed",
              "type": "string"
            },
            "stream_id": {
              "index": "not_analyzed",
              "type": "string"
            },
            "followers_count": {
              "index": "not_analyzed",
              "type": "integer"
            },
            "location": {
              "index": "not_analyzed",
              "type": "geo_point"
            },
            "source_id": {
              "index": "not_analyzed",
              "type": "string"
            },
            "event": {
              "type": "string"
            },
            "screenname": {
              "index": "not_analyzed",
              "type": "string"
            },
            "timestamp": {
              "format": "EE MMM d HH:mm:ss Z yyyy",

```

```

        "type": "date"
      }
    }
  },
  "doc_type_id": "tweet",
  "doc_type_version": "2.0",
  "product_id": "marvel"
},
{
  "mappings": {
    "jstream": {
      "properties": {
        "ll_timestamp": {
          "format": "EE MMM d HH:mm:ss Z yyyy",
          "type": "date"
        },
        "sentiment": {
          "index": "not_analyzed",
          "type": "string"
        },
        "friends_count": {
          "index": "not_analyzed",
          "type": "integer"
        },
        "version": {
          "index": "not_analyzed",
          "type": "string"
        },
        "userid": {
          "index": "not_analyzed",
          "type": "string"
        },
        "twitterid": {
          "index": "not_analyzed",
          "type": "string"
        },
        "url": {
          "index": "not_analyzed",
          "type": "string"
        },
        "stream_id": {
          "index": "not_analyzed",
          "type": "string"
        },
        "followers_count": {
          "index": "not_analyzed",
          "type": "integer"
        },
        "location": {
          "index": "not_analyzed",
          "type": "geo_point"
        },
        "source_id": {
          "index": "not_analyzed",
          "type": "string"
        }
      }
    }
  }
}

```

```

        "event": {
          "type": "string"
        },
        "screenname": {
          "index": "not_analyzed",
          "type": "string"
        },
        "timestamp": {
          "format": "EE MMM d HH:mm:ss Z yyyy",
          "type": "date"
        }
      }
    },
    "doc_type_id": "tweet",
    "doc_type_version": "1.0",
    "product_id": "marvel"
  }
]
}

```

Sample Response with `_flatten`

```

{
  "doc_types": [
    {
      "mappings": {
        "jstream": {
          "properties": {
            "sentiment": {
              "index": "not_analyzed",
              "type": "string"
            },
            "friends_count": {
              "index": "not_analyzed",
              "type": "integer"
            },
            "name.first": {
              "type": "text"
            },
            "name.last": {
              "type": "text"
            }
          }
        }
      }
    },
    "doc_type_id": "nested",
    "doc_type_version": "2.0",
    "product_id": "marvel"
  },
  {
    "mappings": {
      "jstream": {
        "properties": {
          "sentiment": {
            "index": "not_analyzed",
            "type": "string"
          }
        }
      }
    }
  }
]
}

```



```

      "friends_count": {
        "index": "not_analyzed",
        "type": "integer"
      },
      "nested.level1.field1": {
        "type": "geo_point"
      },
      "nested.level1.field2": {
        "type": "long"
      }
    }
  },
  "doc_type_id": "tweet",
  "doc_type_version": "1.0",
  "product_id": "marvel"
}
]
}

```

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```

{
  "_message": "Product not found, provided product_id doesn't exist.",
  "_code": "1002"
}

```

Get Specific Document Type Definition

Request:

URI	HTTP Method
/doc_type(product_id=<product_id>, doc_type_id=<doc_type_id>, doc_type_version=<doc_type_version>)	GET

Directly queries the meta-data index with type="mapping_info" by matching product_id, doc_type_id and doc_type_version fields.

The URL also takes an optional parameter "_flatten," which flattens the mapping response till the leaf node. This returns a flat response for mapping with nested properties. In case of geo-point, the response will be flattened into lat, lon. In case of "nested" type, the response will be flattened until the nested field and it will also be flattened underneath the nested field (see Complex response with nested flattening in Sample response with _flatten example in Get Specific Document Type Definition).

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/doc_type
 (product_id=<product_id>,doc_type_id=<doc_type_id>,doc_type_version=<doc_type_version>)

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	marvel
doc_type_id	A unique identifier for your document type.	Yes	String	tweet
doc_type_version	Version of the document type. Default: 0.0	Yes	String	1.0
_flatten	Flatten the response till type	No	Boolean	_flatten=true _flatten=false

Sample Request

```
GET /doc_type(product_id='marvel', doc_type_id='tweet', doc_type_version='1.0')
```

Response:

Success	201
Error	400

Sample Response

```
{
  "mappings": {
    "data": {
      "properties": {
        "ll_timestamp": {
          "format": "EE MMM d HH:mm:ss Z yyyy",
          "type": "date"
        },
        "sentiment": {
          "index": "not_analyzed",
          "type": "string"
        },
        "friends_count": {
          "index": "not_analyzed",
          "type": "integer"
        },
        "version": {
          "index": "not_analyzed",
          "type": "string"
        },
        "userid": {
          "index": "not_analyzed",
          "type": "string"
        },
        "twitterid": {
          "index": "not_analyzed",
```

```

        "type": "string"
    },
    "url": {
        "index": "not_analyzed",
        "type": "string"
    },
    "stream_id": {
        "index": "not_analyzed",
        "type": "string"
    },
    "followers_count": {
        "index": "not_analyzed",
        "type": "integer"
    },
    "location": {
        "index": "not_analyzed",
        "type": "geo_point"
    },
    "source_id": {
        "index": "not_analyzed",
        "type": "string"
    },
    "event": {
        "type": "string"
    },
    "screenname": {
        "index": "not_analyzed",
        "type": "string"
    },
    "timestamp": {
        "format": "EE MMM d HH:mm:ss Z yyyy",
        "type": "date"
    }
}
}
},
"doc_type_id": "tweet",
"doc_type_version": "1.0",
"product_id": "marvel"
}

```

Sample Response with `_flatten`

```

{
  "mappings": {
    "jstream": {
      "properties": {
        "sentiment": {
          "index": "not_analyzed",
          "type": "string"
        },
        "friends_count": {
          "index": "not_analyzed",
          "type": "integer"
        },
        "name.first": {
          "type": "text"
        }
      }
    }
  }
}

```

```

        "name.last": {
            "type": "text"
        }
    },
    "doc_type_id": "nested",
    "doc_type_version": "2.0",
    "product_id": "marvel"
}

```

Complex response with nested flattening:

```

{
    "mappings": {
        "jstream": {
            "properties": {
                "level1_object.level2_nested": {
                    "level3_obj.level4_field": {
                        "index": "not_analyzed",
                        "type": "string"
                    },
                    "type": "nested"
                },
                "level1_object.level2_object_field1": {
                    "type": "double"
                },
                "level1_object.level2_object_field2.lon": {
                    "type": "double"
                },
                "level1_nested": {
                    "type": "nested",
                    "level2_obj.level3_field": {
                        "index": "not_analyzed",
                        "type": "string"
                    },
                    "level2_field": {
                        "index": "not_analyzed",
                        "type": "string"
                    },
                    "key": "value"
                },
                "level1_object": {
                    "index": "not_analyzed",
                    "type": "object"
                },
                "level1_object.level2_object_field2.lat": {
                    "type": "double"
                },
                "level1_object.level2_object_field2": {
                    "type": "geo_point",
                    "key": "value"
                }
            }
        }
    },
    "doc_type_id": "nested",
    "doc_type_version": "3.0",
}

```

```

    "product_id": "marvel"
  }

```

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message.

Sample Error Response

```

{
  "_message": "Invalid URL. ",
  "_code": "1019"
}

```

Update Document Type

Request:

URI	HTTP Method
/onboarding/doc_type	PATCH

You have already seen how to create (POST) and read (GET) doc_type using the Jarvis APIs. This section explains how you can add new fields (PATCH) to the existing doc_type. You cannot modify or delete the existing fields. A PATCH request adds the new mappings or updates the retention_period for a given combination of product_id, doc_type_id, and doc_type_version.

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/doc_type

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	marvel
doc_type_id	A unique identifier for your document type.	Yes	String	tweet
doc_type_version	Version of the document type.	Yes	String	1.0
unique_id	A valid JSON Path.	Yes. This parameter is mandatory only if it was provided as part of the POST request.	String	['stream_id']
mappings	Information about the data fields that will be ingested.	Yes	JSON structure	See sample request below

Name	Description	Required?	Type	Example
retention_period	The number of days the ingested document should be kept in Jarvis. This value will be used by the Purge service while deleting old documents. If you keep 0 as the retention_period, the Purge service will not delete any ingested data.	No	Integer	10
tenant_doc_type_details	Retention period per tenant for the current doc type.	No	JSON Array	[{"tenant_id": "t2", "retention_period": 2}]

Note: The field marked as "unique_id" should be of type String only.

Assume that a doc_type given in the following code is already created, and you want to update this API to add a field at the top level and one at the geolocation level.

Sample Request

```
{
  "product_id": "DXA",
  "doc_type_id": "WB",
  "doc_type_version": "1.1",
  "unique_id": "${'geolocation'}['timeStamp']",
  "retention_period": 10,
  "mappings": {
    "jstream": {
      "properties": {
        "firstname": {
          "type": "integer"
        },
        "lastname": {
          "type": "string",
          "index": "not_analyzed"
        }
      },
      "geolocation": {
        "type": "object",
        "properties": {
          "timeStamp": {
            "type": "integer"
          },
          "geopoint": {
            "type": "geo_point"
          },
          "altitude": {
            "type": "integer"
          },
          "longitude": {
            "type": "integer"
          }
        }
      }
    }
  }
}
```

```

    },
    "latitude": {
      "type": "integer"
    }
  }
}
}
},
"tenant_doc_type_details": [
  { "tenant_id": "t2", "retention_period": 2 } ]
}

```

While updating the doc_type using PATCH, you can give the complete mapping also, which will look like the following code. The newly added parts are **highlighted**:

```

{
  "product_id": "DXA",
  "doc_type_id": "WB",
  "doc_type_version": "1.1",
  "unique_id": "$['geolocation']['altitudeAccuracy']",
  "mappings": {
    "jstream": {
      "properties": {
        "about": {
          "type": "string",
          "index": "analyzed"
        },
        "geolocation": {
          "properties": {
            "altitudeAccuracy": {
              "type": "integer"
            }
          }
        }
      }
    }
  }
}

```

As shown in the following code, you can give the complete doc_type also. The newly added parts are **highlighted**:

```

{
  "product_id": "DXA",
  "doc_type_id": "WB",
  "doc_type_version": "1.1",
  "unique_id": "$['geolocation']['altitudeAccuracy']",
  "mappings": {
    "jstream": {
      "properties": {
        "firstname": {
          "type": "integer"
        },
        "lastname": {
          "type": "string",
          "index": "not_analyzed"
        },
        "about": {
          "type": "string",
          "index": "analyzed"
        },
        "geolocation": {
          "type": "object",
          "properties": {
            "timeStamp": {
              "type": "integer"
            },
            "geopoint": {
              "type": "geo_point"
            }
          }
        }
      }
    }
  }
}

```

```

        "altitude": {
          "type": "integer"
        },
        "longitude": {
          "type": "integer"
        },
        "latitude": {
          "type": "integer"
        },
        "altitudeAccuracy": {
          "type": "integer"
        }
      }
    }
  }
}

```

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
Error_message	string	A message that explains the cause of the error.
Error_code	Integer	A unique error code used to identify the error message.

```

{
  "error_message": "doc_type_id and doc_type_version doesnt exist for the given product"
  "error_code": 400
}

```

In this case, data is not routed to the Service Layer (Elasticsearch) or to the Batch Layer (HDFS). Use this configuration, if you want data to be routed and stored only in the Speed Layer (Kafka).

Data Ingestion API

This API helps you ingest data (single or bulk volumes) into the Jarvis platform.

Make the following changes based on your environment (SSL or non-SSL) :

SSL	Non-SSL
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.
curl -XGET 'https://localhost:9200/' --cacert <certificate of CA> as the command for querying Elasticsearch.	curl -XGET 'http://localhost:9200/' as the command for querying Elasticsearch.



Before doing onboarding and ingestion using **Postman client** in an SSL-enabled setup, refer [this procedure](#).

Request:

End point:

URI	HTTP Method
/ingestion	POST

Resource URL:

http(s)://<host name or IP address>:<port number>/ingestion

HTTP Headers:

Name	Description	Required?	Type	Supported Values	Example
priority	Priority of the messages to be ingested.	No	String	p1, p2, and p3. It can be up to p9 based on the settings you have done during installation.	curl -XPOST -H 'priority: p3' <ingestion url>

Parameters:

Name	Description	Required?	Type	Example
product_id	A unique identifier for your product.	Yes	String	AXA
tenant_id	A unique identifier the tenant.	Yes	String	Verizon
doc_type_id	A unique identifier for your document type.	Yes	String	events
doc_type_version	Version of the document type.	Yes	String	1.0

Sample Request:

```
{
  "documents": [
    {
      "header": {
        "product_id": "AXA",
        "tenant_id": "Verizon",
        "doc_type_id": "events"
        "doc_type_version": "1.0"
      },
      "body": [
        {
          "sessionid": "edbaa880-2bca-11e6-b7be-0002a5d5c51b",
          "logs": " 16/05/14 21:00:06 ERROR Master: RECEIVED SIGNAL 15:
SIGTERM"
        }
      ]
    }
  ]
}
```

```

    },
    {
      "header": {
        "product_id": "MAA",
        "tenant_id": "Amazon",
        "doc_type_id": "crash",
        "doc_type_version": "1.1"
      },
      "body": [
        {
          "sessionid": "2baa880-2bca-11e6-b7be-0002a5d5c51b",
          "logs": " 18/05/14 21:00:06 ERROR Master: RECEIVED SIGNAL 15:
SIGTERM"
        }
      ]
    }
  ]
}

```

Note: “header” is the common header format. “body” is the list of documents that must be ingested with the same header.

Response:

- Success – 202
- Service unavailable – 503

Data Access Service API (DAS)

The Data Access Service (DAS) APIs of Jarvis help applications access both application data and analytics insights from Jarvis. This section explains the following topics:

- [JSON Data Representation](#)
- [Jarvis Query Language \(JQL\)](#)
- [Filter Operations](#)
- [Functions](#)
- [Grouping Functions](#)
- [Metric Functions](#)
- [Nested Functions](#)
- [Response Formats](#)
 - [Elements in a JSON Response](#)
 - [Sample JSON Response](#)
- [Queryable Metadata](#)
 - [Sample Responses](#)



Examples in the following section assume twitter data in demonstrating API syntax and semantics.

Make the following changes based on your environment (SSL or non-SSL) :

SSL	Non-SSL
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.
curl -XGET 'https://localhost:9200/' --cacert <certificate of CA> as the command for querying Elasticsearch.	curl -XGET 'http://localhost:9200/' as the command for querying Elasticsearch.

Resource URLs:

the following GET requests help you access the required data and analytics insights from Jarvis.



1. <product_id> and <tenant_id> in the path parameters drive the context in which query is executed. Results should contain only data for that product-tenant.
2. Tenant IDs starting with "_" are reserved for Jarvis. For example, Jarvis uses _system or _jarvis for internal purposes.
3. <tenant_id>_all can be used to query across all tenants of a product.
4. <doc_type> in the URL path segment can be a <doc_type_id>_<doc_type_version> or <doc_view_id> as created using Onboarding API.

GET Path	Description	Examples
/<product_id>/<tenant_id> /<doctype>	Addresses a collection of resources	/twitter/ca/tweets /twitter/_all/tweets
/<product_id>/<tenant_id> /<doc_type>(<id>)	Addresses a resource with a ID.	/twitter/ca/tweets ('tweet_id')
/<product_id>/<tenant_id> /<doc_type>/<function>(....)	Gets result(s) by executing a function on a collection resources	/twitter/ca/tweets /group_by(field='hashtag') /twitter/ca/tweets/avg (field='likes')? _filter=hashtags eq '#ca'
/<product_id>/<tenant_id> /<doc_type>?<query_clause>	Gets resources matching a query. Query syntax is explained below in its own section.	/twitter/ca/tweets? _filter=timestamp gt 2016-01-01
/<product_id>/<tenant_id> /<doc_type>/<function>(....) /<sub_function>(...)/...	Nested execution of functions (apply a function on results of a previously executed function.)	/twitter/ca/tweets /groupby_value (field='hashtags')/avg (field='likes')

Response:

Client-related errors	4xx
Server-related failures	400

JSON Data Representation

Datatype	Description	Default Value	Example
Boolean	Boolean datatype stores either true or false value. This datatype is case-sensitive and should always be in lower case.	false	true
Integer /Long	Integer datatype stores 32-bit numbers that can be positive, negative, or zero. The range is -2147483648 to 2147483647. Long datatype stores 64-bit numbers that can be positive, negative, or zero. The range is -9223372036854775808 to 9223372036854775807 .	0	12345
Float /Double	Float datatype stores 32-bit (7 digits) numbers. The range is $\pm 1.4\text{E}-45$ to $\pm 3.4028235\text{E}+38$. Double datatype stores 64-bit (15-16 digits) numbers. The range is $\pm 4.9\text{E}-324$ to $\pm 1.7976931348623157\text{E}+308$.	0.0	3.1415926535897931
String	String datatype stores character sequence, which can be alphanumeric or special characters, inside single quotes.		'Say Hello'
DateTime	DateTime datatype stores ISO8601 Format for DateTime to represent Date or DateTime.		2012-12-03T07:16:23Z
Geolocation	Geolocation datatype stores Geo Coordinates inside single quotes in 'latitude, longitude' format. Latitude and Longitude are represented as Double types.		'-71.34, 41.12'
Null	Missing or JSON null value.		null

Jarvis Query Language (JQL)

The Data Access REST APIs support the following query parameters:

Query	Description	Example
_skip	Shows results after skipping the first 'n' results.	http://jarvis.ca.com/das/twitter/ca/tweet?_skip=10
_top	Shows the top 'n' results. Precedence: _top is always applied after _skip, if present.	http://jarvis.ca.com/das/twitter/ca/tweet?_top=10
_after	Used in conjunction with _orderby. Shows results after defined values of _orderby fields.	Page 1 Query: http://jarvis.ca.com/das/twitter/ca/tweet?_orderby=user_id & _top=100



Alternative to _skip

Query	Description	Example
	<p><code>_skip + _top</code> cannot be more than 10000 records. Values beyond 10000 throw an error.</p> <p>You can get the <code>_top</code> "N" records first in sorted order by using <code>_orderby</code> and you can get the next batch of results by using the last record's values of <code>_orderby</code> fields as <code>_after</code> marker to get the next batch of results.</p>	<p>Assume that the last record in the previous page has <code>user_id</code> as '@MikeGregoireCA'</p> <p>then for Page 2 Query:</p> <p>http://jarvis.ca.com/das/twitter/ca/tweet?_orderby=user_id&_after=@MikeGregoireCA&_top=100</p>
<code>_search</code>	<p>Free text search.</p> <p>Precedence: <code>_search</code> is always applied after <code>_filter</code>, if present.</p>	<p>http://jarvis.ca.com/das/twitter/ca/tweet?_search=jarvis 2.0</p>
<code>_filter</code>	Filters expression that the results should match.	<p>http://jarvis.ca.com/das/twitter/ca/tweet?_filter=(name eq 'jarvis' and version lt '2.0') or not (version gt '3.0')</p>
<code>_orderby</code>	Sorts order of the results.	<p>http://jarvis.ca.com/das/twitter/ca/tweet?_orderby=name, likes desc, city</p>
<code>_select</code>	Shows the list of properties to show in response.	<p>http://jarvis.ca.com/das/twitter/ca/tweet?_select=name, likes, retweets, tags</p>
<code>_flatten</code>	<p>Flattens the aggregation buckets from a tree-like structure to an array of objects. Note that this query is applicable only on results that are grouped using the <code>groupby</code> function.</p> <p>From Jarvis 2.1, <code>_flatten</code> applies to documents returned in the response from DAS. It applies to documents returned as a result of any query. A sample DAS response with and without flatten has been attached to this wiki page.</p>	<p>http://jarvis.ca.com/das/twitter/ca/tweet/groupby_value(field='hashtag')?_flatten=true</p> <p>http://jarvis.ca.com/das/twitter/ca/tweet/groupby_value(field='hashtag')?_flatten</p>

Here is a sample query:

```
http://jarvis.ca.com/das/axa/verizon/appflow? filter=key eq regex('DEA5B6C5-3502-F069-7584-2E132721785F'#"AngryBirds"#"1.0"#"iOS') and rf eq
1&_select=key&_orderby=key desc&_top=1000
```



You can query DAS to match fields that are having missing fields.

Example:

http://jarvis.ca.com/das/twitter/ca/tweet?_filter=username ne null

http://jarvis.ca.com/das/twitter/ca/tweet?_filter=username eq null

Filter Operations

Operator	Description	Example
Comparison Operators		
eq	Equal	user_name eq 'jarvis'
ne	Not equal	user_name ne regex([a-z]*)
gt	Greater than	likes gt 20
ge	Greater than or equal	likes ge 10
lt	Less than	likes lt 20
le	Less than or equal	likes le 100
in	In a range of values	location in georadius('10.0,10.0', '100km')
Logical Operators		
and	Logical and. Both operands must evaluate to true.	retweets le 200 and likes gt 50
or	Logical or. One of the expressions must evaluate to true.	tags has 'java' or tags has 'javascript'
not	Logical negation. Flips the boolean value of the expression.	not (contains(name,'jarvis'))
Grouping Operators		
()	Precedence grouping. Default precedence is AND, OR. Defines the order in which expressions are evaluated by building a parse tree for each ().	(retweets le 200 and likes gt 50) or (not (contains(name,'jarvis')))
String Functions		
regex match	Regex Match on string fields	name eq regex([a-z]*) Note: Closing parenthesis inside a regular expression must be escaped using double single quotes. Example: _filter=name1 eq regex(j~`!@#\$%^&*(")_+=?><./\"';\".*s) Lucene Regex Engine evaluates regular expressions.
Geo Functions		
Function	Description	Example
geo radius	Match documents in specified radius from a given geo location.	georadius('10.0,75.1', '100km')

geo range	Match documents between a specified range from geo location.	georange('-10.0,75.1', '10km', '100km')
geo boundary	Match documents that fall in a geo box from specified top left to bottom right geo locations	geoboundary ('-10.0,75.1', '10.0,85.1')

Functions

Functions are special operations performed on a resource collection or on a specific resource. By that definition, aggregates can be treated as functions.

Examples: Group Tweets by Number of Likes. or Group Tweets by Hashtags.

Jarvis supports functions as first class REST operations with an intuitive syntax. Functions are executed through GET call and are idempotent.

Example URL	Description
/tweets/groupby_value(field='retweets')	Group by value of the field "retweets"

Grouping Functions

Grouping functions group (buckets) the results based on an input criteria. Jarvis supports the following grouping functions.



Aggregation Name

- All aggregate functions optionally take "name" parameter that can be used to specify the name of the aggregation in response.
- Values of type "Boolean" are returned as integer 0 or 1 in the aggregation response for grouping functions.

Function Name	Syntax and Examples	Function Parameters	Description
groupby_value	<p>/tweets/groupby_value(field='tags')</p> <p>Ordering based on sub-aggregates:</p> <p>/tweets/groupby_value(field='tags', orderby='avg desc', top=2)/avg(field='likes')</p> <p>/tweets/groupby_value(field='tags', orderby='avg_likes')/avg(field='likes', name='avg_likes')</p> <p>Note: orderby can be applied only if the groupby is followed by a subaggregate metric function.</p>	field and top	Group results by value of a specific field
groupby_timeinterval			Group results by time duration/period

Function Name	Syntax and Examples	Function Parameters	Description
	<pre>/tweets/groupby_timeinterval (field='creation_date', interval='day', timezone='+05:30')</pre> <pre>/tweets/groupby_timeinterval (field='creation_date', interval='day_of_week', timezone='+05:30')</pre>	field, interval, timezone	<p>Following are the supported intervals:</p> <p>day, month, year, hour_of_day, day_of_week, and month_of_year.</p>
groupby_interval	<pre>/tweets/groupby_interval (field='likes', interval=1000)</pre>	field, interval	Group results by numeric interval
groupby_range	<pre>/Tweets/groupby_range (field='comment_count', normal: range=(,10), popular:range=(10,20), very_popular:range=(20,))</pre>	field, range	<p>Group results by range of values.</p> <p>Optionally, names of each range are assigned by the prefix 'name:' convention.</p>
groupby_filter	<pre>/Pages/groupby_filter(java: filter=tags eq 'java', javascript: filter=tags eq 'javascript')</pre>	filter	<p>Group results by matching filters.</p> <p>Optionally, names of each filter results are assigned by the prefix 'name:' convention.</p>
groupby_georange	<pre>/Tweets/groupby_georange (field='location', origin='lat,lon', unit='km', near:range=(,10), far: range=(50,))</pre>	field, origin, unit, and range	<p>Group results by geo location distance range from a specified point of origin</p> <p>Optionally, names of each filter results are assigned by the prefix 'name:' convention.</p>

Metric Functions

Aggregate functions compute an aggregate function on the result set and return a value. Jarvis will support the following aggregate functions.

Function Name	Syntax	Description
min	<pre>/Tweets/min (field='comment_count')</pre>	Min of comment_count
max	<pre>/Tweets/max (field='comment_count')</pre>	Max of comment_count
avg	<pre>/Tweets/avg (field='comment_count')</pre>	Average of comment_count
sum	<pre>/Tweets/sum (field='comment_count')</pre>	Sum of comment_count
ext_stats		

Function Name	Syntax	Description
	/Tweets/ext_stats (field='comment_count')	Extended statistics on comment_count (min, max, avg, sum, mean..)
stats	/Tweets/stats (field='comment_count')	Basic Statistics on comment_count (min, max, avg, sum)
count	/Tweets/count (field='comment_count')	Count of number of entries of comment_count



Supported Functions

These are only the initial set of functions supported. New functions will be added from time to time based on requirements.

Nested Functions

Functions are expected to return results by evaluating a function on a resource collection. A second-level function can be further executed on the results of the previous function.

This is accomplished by nesting functions in the REST URL.



DAS limits the number of levels of nested aggregates to 4 by design.

Example URL	Description
/tweets/groupby_value(field='retweets')/avg (field='age')	Here an aggregation of group by "retweets" value and a sub
/tweets/groupby_value(field='retweets', orderby='avg',)/avg(field='age')	

Response Formats

Elements in a JSON Response

Element	Description
results	Result of the query. This element is always part of the response.
results. doc_count	Total number of documents matching the query criteria. This element is always part of the response.
results. next_page	Link to fetch the next page of results if "documents" contains paged results. This element is present in the response when DAS paginates the documents in result set. This includes a _skiptoken parameter that can be used by clients to get the next batch of results. The _skiptoken parameter is typically valid for a few minutes as configured for DAS.

Element	Description
results. documents	List of matching documents (what you would see inside <code>_source</code> in ES). Aggregate queries without <code>_top</code> query do not contain matching documents.
aggregations	Aggregations in the response. This element is present for aggregate queries.

Sample JSON Response

```
{
  "results":
  {
    "doc_count": 142,
    "next_page" : "<href>",
    "documents":
    [
      {
        "tenant_id": "Verizon",
        "sentiment": "neutral",
        "friends_count": "1997",
        ...
      },
      {
        "tenant_id": "Verizon",
        "sentiment": "negative",
        "friends_count": "189",
        ...
      }
    ]
  },
  "aggregations":
  {
    "groupby_range_friends_count":
    {
      "buckets":
      [
        {
          "doc_count": 89588,
          "key": "normal",
          "to": 100
        },
        {
          "doc_count": 1729,
          "from": 100,
          "key": "popular",
          "to": 1000
        },
        ...
      ]
    }
  }
}
```

Queryable Metadata

To get metadata for the response format for a DAS query, `_metadata` can be applied to the query. This returns only the metadata. The format adheres to the mapping scheme that is used in onboarding a doc type.



To get metadata format for a query http://jarvis.ca.com/das/product/tenant/doctype?_flatten=true, user can query http://jarvis.ca.com/das/product/tenant/doctype?_metadata. Note that for the same query, `_metadata` has been appended

Sample Responses

The sample queries for the metadata responses are based the mappings shown in the following section in the properties field of `results.documents`:

```
{
  "mappings": {
    "jstream": {
      "properties": {
        "results": {
          "type": "object",
          "properties": {
            "doc_count": {
              "type": "long"
            },
            "documents": {
              "type": "nested",
              "properties": {
                "count": {
                  "type": "integer"
                },
                "id": {
                  "type": "string"
                },
                "location": {
                  "type": "geo_point"
                },
                "timestamp": {
                  "format": "yyyy-MM-dd'T'HH:mm:ssZ",
                  "type": "date"
                }
              }
            }
          }
        }
      }
    }
  },
  "doc_type_id": "doctype",
  "doc_type_version": "ver",
  "product_id": "product"
}
```

DocView API

Jarvis deals with the metadata of the ingested documents using Doc Types. Each Doc Type is uniquely identified using the `doc_type_id` and `doc_type_version` combination. Each Doc Type is provided its own mapping to define the list of fields and their data types. Both ingestion and access APIs deal with write and querying on single doc types which is in **`doc_type_id_doc_type_version`** format. However, there may be cases where a Jarvis users want to query or aggregate across multiple doc types. To address such requirements, Jarvis enables "Doc Views".

Doc View is a logical name given to a set of Doc Types of a Product to enable searching across multiple Doc Types. A Doc View is analogous to a database view, which is a searchable object in a database that is defined by a query. CRUD on Doc Views are supported on the Jarvis Onboarding service.

Make the following changes based on your environment (SSL or non-SSL) :

SSL	Non-SSL
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.
curl -XGET 'https://localhost:9200/' --cacert <certificate of CA> as the command for querying Elasticsearch.	curl -XGET 'http://localhost:9200/' as the command for querying Elasticsearch.

Create a Doc View

Request:

End point:

URI	HTTP Method
/doc_view	POST


Resource URL:

http(s)://<host name or IP address>:8080/onboarding/doc_view

Sample Request:

```
content-type: application/json
{
  "product_id" : "axa",
  "doc_view_id" : "apache_logs",
  "doc_types" : ["apache_commons_1.0", "apache_combined_1.0"]
}
```

Field	Description	Type	Needed?	Example
product_id	A unique identifier for your product.	String	Yes	ERP
doc_view_id		String	Yes	Apache_Logs

Field	Description	Type	Needed?	Example
	A unique identifier for your document type.			
 Provide either the doc_types or doc_type_patterns . Providing both will result in an error.				
doc_types	An array of different document types. It must be in the following format: doc_type_id+_+doc_type_version.	String	Yes (if doc_type_patterns 0 is not specified)	['apache_comme
doc_type_patterns	An array of different document type patterns. Provide either the doc_types or doc_type_patterns. Providing both will result in an error	String	Yes (if doc_types is not specified)	['tweet_.*', 'facebook_.*']

Response:

No Content–204

Error:

Conflict–409

Bad Request–400

Not Found–404

Update a Doc View

Request:

End point:

URI	HTTP Method
/doc_view	PUT

Resource URL:

http(s)://<host name or IP address>:8080/onboarding/doc_view(product_id="", doc_view_id=")

Sample Request:

```

content-type: application/json
{
  "product_id" : "axa",
  "doc_view_id" : "apache_logs",
  "doc_types" : [ "apache_commons_1.0", "apache_combined_1.0" ]
}

```

Response:

No Content–204

Error:

Bad Request–400

Not Found–404

Get a Doc View

Request:

GET http://jarvis.ca.com/onboarding/doc_view(product_id='axa', doc_view_id='apache_logs')

Response:

Created–200

```
content-type: application/json
{
  "product_id" : "axa",
  "doc_view_id" : "apache_logs",
  "doc_types" : ["apache_commons_1.0", "apache_combined_1.0"]
}
```

Error:

Bad Request–400

Not Found–404

Get all Doc Views for a Product

Request:

GET http://jarvis.ca.com/onboarding/doc_view(product_id='axa')

Response:

Created–200

```
content-type: application/json
{
  "doc_views" : [
    {
      "product_id" : "axa",
      "doc_view_id" : "apache_logs",
      "doc_types" : ["apache_commons_1.0", "apache_combined_1.0"]
    },
    ....
  ]
}
```

Error:

Bad Request—400

Not Found—404

Delete a Doc View**Request:**DELETE http://jarvis.ca.com/onboarding/doc_view(product_id='axa', doc_view_id='apache_logs')**Response:**

No Content—204

Error:

Bad Request—400

Not Found—404

Speed Job Onboarding API

- [Onboard a Speed Job](#)
- [Update a Speed Job](#)

This API helps you onboard a speed job into CA Jarvis and schedule it to run either immediately or at a specified time so that you can visualize near real-time insights of the data.

Make the following changes based on your environment (SSL or non-SSL) :

SSL	Non-SSL
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.
curl -XGET 'https://localhost:9200/' --cacert <certificate of CA> as the command for querying Elasticsearch.	curl -XGET 'http://localhost:9200/' as the command for querying Elasticsearch.

Onboard a Speed Job

Request:

URI	HTTP Method
/ speedjob	POST

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/ speedjob

Parameters:

Name	Description	Required?	Type	Example
jobs_zip	Contains the zip of the speed job that should be submitted to CA Jarvis.	Yes	file	path\job1.zip
content	Contains job-related details.	Yes	String	[{ "key": "content", "value": "{\n \"job_id\": \"pi9\", \n \"product_id\": \"product1\", \n \"job_description\": \"This is a stream job\", \n \"in_doctype\": [{\n \"doc_type_id\": \"indoc_type_id\", \n \"doc_type_version\": \"1.0\" \n }], \n \"out_doctype\": [{\n \"doc_type_id\": \"outdoc_type_id\", \n \"doc_type_version\": \"1.0\" \n }], \n \"job_zip\": \"location/to/hadoop/zip/\", \n \"job_config\": {\"mainclass\": \"pi.py\", \"args\": []}, \n \"scheduler\": {\"when\": \"2017-10-09T17:08:00\" \n }, \n \"microbatch_interval\": \"20\", \n \"batchjob_id\": \"12\" \n }, \"description\": \"\" }]
	job_id: A unique identifier for the speed job.	Yes	String	
	product_id: identifier of the product with which the job is associated.	Yes	String	
	job_description: Brief description about the speed job.	Optional	String	
	job_language: The language the job is implemented in.	Optional	String	\"job_description\": \"java\"
	in_doctype: The list of doctype that forms the input for the speed job. These doctypes must have already been onboarded for the product.	Yes	List of Json	\"in_doctype\": [{\"doc_type_id\": \"hwddoctype\", \"doc_type_version\": \"2.0\"}],
	out_doctype: The list of doctype that forms the output of the speed job. These	Yes	List of Json	\"out_doctype\": [{\"doc_type_id\": \"hwddoctype\", \"doc_type_version\": \"2.0\"}],

Name	Description	Required?	Type	Example
	doctypes must have already been onboarded for the product.			
job_config:	Contains config details for the given speed job.	Yes	Json	"job_config": {"mainclass": "\pi.py", "args": []},
	<ul style="list-style-type: none"> mainclass: The Python main class . (String) args: The list of arguments to the Python job. (List of strings) 			
scheduler:	Contains the scheduling details for the job.	Yes	Json	To run immediately: "scheduler": {"when": "now"} To run at a specific time: "scheduler": {"when": "2017-10-18T18:50:00"}
	<ul style="list-style-type: none"> when: Specify now to run the job immediately, or provide a valid java date-time value to schedule the job. (date time) 			
microbatch_interval:	Microbatch interval time for the speed job. Default: 60 Seconds	Optional	Integer	"microbatch_interval": 30
batchjob_id:	The id of the batch job whose model should be available to the speed job.	Optional		

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message

Update a Speed Job

Request:

URI	HTTP Method
/ speedjob	PATCH

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/ speedjob

Parameters:

Name	Description	Required?	Type	Example
jobs_zip	Contains the zip of the speed job that should be submitted to CA Jarvis.	Optional	file	path\job1.zip
content	Contains job-related details.	Yes	String	[{ "key": "content", "value": "{\n \"job_id\": \"pi9\", \n \"product_id\": \"product1\", \n \"job_description\": \"This is a stream job\", \n \"in_doctype\": [{\n \"doc_type_id\": \"indoc_type_id\", \n \"doc_type_version\": \"1.0\" \n }], \n \"out_doctype\": [{\n \"doc_type_id\": \"outdoc_type_id\", \n \"doc_type_version\": \"1.0\" \n }], \n \"job_zip\": \"location/to/hadoop/zip/\", \n \"job_config\": {\"mainclass\": \"pi.py\", \"args\": []}, \n \"scheduler\": {\n \"when\": \"2017-10-09T17:08:00\" \n }, \n \"microbatch_interval\": \"20\", \n \"batchjob_id\": \"12\" \n }, \"description\": \"\" }]
job_id:	A unique identifier for the speed job.	Yes	String	
product_id:	identifier of the product with which the job is associated.	Yes	String	
job_description:	Brief description about the speed job.	Optional	String	
job_language:	The language the job is implemented in.	Optional	String	\"job_description\": \"java\"
in_doctype:	The list of doctype that forms the input for the speed job. These doctypes must have	Optional	List of Json	\"in_doctype\": [{\"doc_type_id\": \"hwddoctype\", \"doc_type_version\": \"2.0\"}],

Name	Description	Required?	Type	Example
	already been onboarded for the product.			
	out_doctype: The list of doctype that forms the output of the speed job. These doctypes must have already been onboarded for the product.	Optional	List of Json	"out_doctype": [{"doc_type_id": "hwdoctype", "doc_type_version": "2.0"}],
	job_config: Contains config details for the given speed job. <ul style="list-style-type: none"> mainclass: The Python main class . (String) args: The list of arguments to the Python job. (List of strings) 	Optional	Json	"job_config\": {\\"mainclass\\":\\"pi.py\\",\\"args\\":[]},
	scheduler: Contains the scheduling details for the job. <ul style="list-style-type: none"> when: Specify now to run the job immediately, or provide a valid java date-time value to schedule the job. (date time) 	Optional	Json	To run immediately: "scheduler": {"when": "now"} To run at a specific time: "scheduler": {"when": "2017-10-18T18:50:00"}
	microbatch_interval: Microbatch interval time for the speed job. Default: 60 Seconds	Optional	Integers	"microbatch_interval": 30
	batchjob_id: The id of the batch job whose model should be available to the speed job.	Optional		

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message

Batch Job Onboarding API

- [Onboard a Batch Job](#)
- [Update a Batch Job](#)
- [Cron Expressions](#)
 - [Special Characters:](#)
 - [More Examples of Cron-Expressions:](#)

This API helps you onboard a batch job into CA Jarvis and schedule it to run either immediately or at a specified time so that you can visualize near real-time insights of the data.

Make the following changes based on your environment (SSL or non-SSL) :

SSL	Non-SSL
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.
curl -XGET 'https://localhost:9200/' --cacert <certificate of CA> as the command for querying Elasticsearch.	curl -XGET 'http://localhost:9200/' as the command for querying Elasticsearch.



Before doing onboarding and ingestion using **Postman client** in an SSL-enabled setup, refer [Troubleshooting](#).

Onboard a Batch Job

Request:

URI	HTTP Method
/batchjob	POST

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/batchjob

Parameters:

Name	Description	Required?	Type	Example
job_zip	Contains the zip of the batch job that should be submitted to CA Jarvis.	Yes	file	path\job1.zip
content	Contains job-related details.	Yes	String	[<pre> { "key": "content", "value": "{\n \"job_id\": \"pi9\", \n \" product_id\": \"product1\", \n \" job_description\": \"This is a stream job\", \n \" in_doctype\": [{\n \"doc_type_id\": \" indoc_type_id\", \n \"doc_type_version\": \"1.0\" \n }], \n \"out_doctype\": [{\n \"doc_type_id\": \" outdoc_type_id\", \n \"doc_type_version\": \"1.0 \n }], \n \"job_config\": {\"mainclass\": \"pi. py\", \"args\": []}, \n \"scheduler\": {\n \"cron\": \" 0 0/1 * * * ?\"} }] </pre>
	job_id: A unique identifier for the batch job.	Yes	String	
	product_id: Identifier of the product with which the job is associated.	Yes	String	
	job_description: Brief description about the batch job.	Optional	String	
	in_doctype: The list of doctype that forms the input for the batch job. These doctypes must have already been onboarded for the product.	Yes	List of Json	"in_doctype": [{"doc_type_id": "hwdoctype", "doc_type_version": "2.0"}],
	out_doctype: The list of doctype that forms the output of the batch job. These doctypes must have already been onboarded for the product.	Yes	List of Json	"out_doctype": [{"doc_type_id": "hwdoctype", "doc_type_version": "2.0"}],
	job_config: Contains configuration details for the given batch job.	Yes	Json	"job_config": {"mainclass": "pi.py", "args": []},

Name	Description	Required?	Type	Example
	<ul style="list-style-type: none"> mainclass: The Python main class. (String) args: The list of arguments to the Python job. (List of strings) 			
	scheduler: Contains the scheduling details for the job. cron: values: now or a valid java date-time. Refer Cron Expressions for more information.	Yes	Json	To fire the job in every minutes: cron\:\"0 0/1 * * * ?\"

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message

Update a Batch Job

Request:

URI	HTTP Method
/batchjob	PATCH

Resource URL:

http(s)://<host name or IP address>:<port number>/onboarding/batchjob

Parameters:

Name	Description	Required?	Type	Example
job_zip	Contains the zip of the batch job that should be submitted to CA Jarvis.	Optional	file	path\job1.zip
content	Contains job-related details.	Yes	String	[{ "key": "content",

Name	Description	Required?	Type	Example
				<pre> "value": "{\n \"job_id\": \"pi9\", \n \" product_id\": \"product1\", \n \" job_description\": \"This is a stream job\", \n \" in_doctype\": [{\n \"doc_type_id\": \" indoc_type_id\", \n \"doc_type_version\": \"1.0\" \n }], \n \"out_doctype\": [{\n \"doc_type_id\": \" outdoc_type_id\", \n \"doc_type_version\": \"1.0 \n\n }], \n \"job_config\": {\"mainclass\": \"pi. py\", \"args\": []}, \n \"scheduler\": {\n \"{cron\": \" 0 0/1 * * * ?\"} }] </pre>
	job_id: A unique identifier for the batch job.	Yes	String	
	product_id: Identifier of the product with which the job is associated.	Yes	String	
	job_description: Brief description about the batch job.	Optional	String	
	in_doctype: The list of doctype that forms the input for the batch job. These doctypes must have already been onboarded for the product.	Optional	List of Json	"in_doctype": [{"doc_type_id": "hwdoctype", "doc_type_version": "2.0"}],
	out_doctype: The list of doctype that forms the output of the batch job. These doctypes must have already been onboarded for the product.	Optional	List of Json	"out_doctype": [{"doc_type_id": "hwdoctype", "doc_type_version": "2.0"}],
	job_config: Contains configuration details for the given batch job.	Optional	Json	"job_config": {"mainclass": "pi.py", "args": []},
	<ul style="list-style-type: none"> mainclass: The Python main class. (String) args: The list of arguments to the Python job. (List of strings) 			
		Optional	Json	To fire the job in every minutes:

Name	Description	Required?	Type	Example
	<p>scheduler: Contains the scheduling details for the job.</p> <p>cron: values: now or a valid java date-time. Refer Cron Expressions for more information.</p>			cron\":"0 0/1 * * * ?\"

Response:

Success	204
Error	400

Error Response:

Property	Type	Description
_message	string	A message that explains the cause of the error.
_code	Integer	A unique error code used to identify the error message

Cron Expressions

These expressions define a schedule. Cron-Expressions are strings that are made up of seven sub-expressions, that describe individual details of the schedule. These sub-expressions are separated with white-space. The sub-expressions represent the following:

Field	Allowed Values	Allowed Special Characters
Seconds	0 to 59	, - * /
Minutes	0 to 59	, - * /
Hours	0 to 23	, - * /
Day-of-Month	1 to 31	, - * ? / L W
Month	0 to 11 or JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, and DEC	, - * /
Day-of-Week	1 to 7 (1 = Sunday) or SUN, MON, TUE, WED, THU, FRI, and SAT	, - * ? / L #
Year (optional)	1970-2099	, - * /

Special Characters:

Character	Description	Example
,	To specify additional values.	MON, WED, FRI.
-	To specify ranges.	MON-WED.

Character	Description	Example
*	To specify all values with in a field.	* in the place of the month field in the previous example indicates every month.
/	To specify increments.	0/12 in the seconds field means the seconds 0, 12, 24, 36, and so on.
?	To specify no specific value or to override the field.	To schedule something for a particular day of the month (say, the 20th), when it does not matter what day of the week that happens to be, put 20 in the day-of-month field, and ? in the day-of-week field.
L	<ul style="list-style-type: none"> In the day-of-month field, it means the last day of the month. If used in the day-of-week field along with no other value, it means Saturday. If used in the day-of-week field along with another value (X), it means the last X day of the month. <p>Note: Do not use lists, or ranges of values along with L to avoid confusing results.</p>	2L in the day-of-week field means the last Monday of the month.
W	To specify the weekday (Monday-Friday) nearest to the given day.	12W as the value for the day-of-month field indicates the nearest weekday to the 12th of the month. So if the 12th is a Saturday, Friday the 11th is selected. If the 12th is a Sunday, Monday the 13th is selected. If the 12th is a Wednesday, Wednesday the 12th is selected.
#	To specify the nth day of the month.	MON#3 or 2#3 in the day-of-week field indicates the third Monday of the month.

More Examples of Cron-Expressions :

Expression	Meaning
0 0 11 * * ?	Fire at 11:00 AM every day.
0 25 10 ? * *	Fire at 10:25 AM every day.
0 25 10 * * ?	Fire at 10:25 AM every day.
0 25 10 * * ? *	Fire at 10:25 AM every day.
0 25 10 * * ? 2017	Fire at 10:25 AM every day starting from the year 2017.
0 10 10 10 10 ?	Fire every October 10 at 10:10 AM.

Expression	Meaning
0 * 15 * * ?	Fire every minute starting at 3:00 PM and ending at 3:59 PM, every day.
0 0/5 15 * * ?	Fire every 5 minutes starting at 3:00 PM and ending at 3:55 PM, every day.
0 0/5 15,20 * * ?	Fire every 5 minutes starting at 3:00 PM and ending at 3:55 PM, AND fire every 5 minutes starting at 8:00 PM and ending at 8:55 PM, every day.
0 0-5 13 * * ?	Fire every minute starting at 1:00 PM and ending at 1:05 PM, every day.
0 10,20 13 ? 6 WED	Fire at 1:10 PM and at 1:20 PM every Wednesday in the month of June.
0 30 11 ? * MON-FRI	Fire at 11:30 AM every Monday, Tuesday, Wednesday, Thursday, and Friday.
0 30 11 10 * ?	Fire at 11:30 AM on the 10th day of every month.
0 30 11 L * ?	Fire at 11:30 AM on the last day of every month.
0 30 11 ? * 4L	Fire at 11:30 AM on the last Wednesday of every month.
0 30 11 ? * 2L	Fire at 11:30 AM on the last Monday of every month.
0 30 11 ? * 6L 2015-2020	Fire at 11:30 AM on every last friday of every month during the years 2015 and 2020.
0 30 11 ? * 6#2	Fire at 11:30 AM on the second Friday of every month.
0 0 15 1/3 * ?	Fire at 3 PM every 3 days every month, starting on the first day of the month.

Health Check API

The health of different CA Jarvis components can be checked through a RESTful API. These APIs enable integrating Jarvis health check into any monitoring dashboard.

Make the following changes based on your environment (SSL or non-SSL):

SSL	Non-SSL
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.

Check the Health of All the Jarvis Services

Request:

URI	HTTP Method
/health	GET

This API checks the status of each service of CA Jarvis, and displays the following responses:

- "green" : Service is healthy.
- "red": Service is unhealthy.

Resource URL:

http(s)://<host name or IP address>:<port number>/health

Response:

Success (Green)	200
Error (Red)	200

```
{
  "services":[
    {
      "onboarding":{
        "status":"green"
      }
    },
    {
      "ingestion":{
        "status":"green"
      }
    },
    {
      "das":{
        "status":"green"
      }
    },
    {
      "verifier":{
        "status":"green"
      }
    },
    {
      "indexer":{
        "status":"green"
      }
    },
    {
      "kafka":{
        "status":"green"
      }
    },
    {
      "schema_registry":{
        "status":"green"
      }
    },
    {
      "elasticsearch":{
        "status":"green"
      }
    },
    {
      "kron":{
        "status":"green"
      }
    },
    {
      "utils":{
        "status":"green"
      }
    }
  ],
  "status":"green"
}
```

Check the Health of Individual Services

Request:

URI	HTTP Method
/health/<name of the service>	GET

This API checks the status of the onboarding services of CA Jarvis, and displays the following responses:

- "green" : Service is healthy.
- "red": Service is unhealthy.

Resource URL:

http(s)://<host name or IP address>:<port number>/health/<name of the service>

<name of the service> accepts one of the following values:

- onboarding
- ingestion
- das
- verifier
- indexer
- elasticsearch
- kafka
- schema_registry
- kron
- utils

Response:

Healthy (Green)	200
Unhealthy (Red)	200

Sample Output for Onboarding Service:

```
{
  "onboarding": {
    "status" : "green"
  }
}
```

CA Jarvis Dashboard

The CA Jarvis Dashboard helps you visualize real-time analytics by creating comprehensive business reports. CA Jarvis Dashboard is a visualization platform that is designed to search, view, and interact with the data that is stored in CA Jarvis.

- [Components](#)
- [User Interface](#)
- [Using Viewlets](#)
- [Using Dashboards](#)

- [API Reference of CA Jarvis Dashboard](#)

Prerequisites for Using CA Jarvis Dashboard

- Ensure that you have installed CA Jarvis.
- Ensure that you have installed CA Jarvis Dashboard.

Accessing CA Jarvis Dashboard

Customize and use the following URL to access CA Jarvis Dashboard:

`http(s)://<IP of the host where is installed>:<Port number. 8080 (non-SSL), 8443 (SSL)>`

Example:

`http://58.225.6.97:8080`

You must create a user profile to access CA Jarvis Dashboard. Provide the following information to create a user profile:

- Your valid email address (Used as your username)
- Product ID (Case-sensitive) Check with your administrators.
- Tenant ID (Case-sensitive) Check with your administrators.
Note: A product ID can have multiple Tenant IDs associated with it.
- Password (Case-sensitive)



Note: The username must be unique for a given combination of Product ID and Tenant ID.

If you are using CA Jarvis Dashboard for the first time, click **SIGN UP**, and follow the prompts.

If you have already created your profile, click **SIGN IN**, and follow the prompts.

If you have forgotten your password, click the **FORGOT PASSWORD?** link and follow the prompts. The system sends a unique URL to your email address. The URL expires in 48 hours, but the period can be configured. You can use the URL to create a password.

Components

CA Jarvis Dashboard offers the following components to visualize your data:

- [Viewlets](#)
- [Dashboards](#)
- [Layout Templates](#)
- [Charts](#)
 - [Common Charts](#)

- [Advanced Charts](#)
 - [Geographical Heatmaps](#)
 - [Key Metric Chart](#)
 - [Scatter Chart](#)
 - [Bubble Chart](#)
 - [Outlier Charts](#)
 - [Cluster Charts](#)
 - [Progress Bar Charts](#)
 - [Heat Table](#)
 - [Box and Whiskers Chart](#)

Viewlets

Viewlet visualizes the data that is queried from CA Jarvis in the form of charts and graphs.

Viewlet objects are private to the creators of the Viewlet. You can share your Viewlets with other users that fall under the same tenant. The recipients cannot change the shared Viewlet, but they can duplicate it. The duplicated objects are characteristically identical to the original object and the ownership falls under the user who has duplicated it.

Dashboards

Viewlets are added, and viewed, through dashboards. Dashboards can contain many viewlets. A viewlet can be associated with multiple dashboards, and a user can create many dashboards. Dashboards are private to the creators of the dashboard.

Layout Templates

The Layout Templates helps you to select a layout that is based on your requirement. For example, if you want to get insights into the high-level key performance indicators (KPIs) you can select High-Level Reporting Dashboard as the template. In a dashboard, you must associate viewlets with panels to populate the dashboard with charts and data grids. You can either create a viewlet, or drag-and-drop existing viewlets to the panels in the dashboard. Creators can add many rows to a dashboard. Adding rows to the dashboards does not change the structure of the original layout template. The system provides five predefined layout templates. The system also has a fully customizable template, which contains no panels. The user can create any panel arrangement (permissible within the design of the system) for a given dashboard.

Charts

Charts help in visualizing the data from Jarvis. You can use common charts or advanced charts while creating viewlets. The following chart types are currently supported:

Common Charts

- Table chart
- Vertical bar chart
- Horizontal bar chart
- Pie chart
- Donut pie chart

- Line chart
- Area chart
- Stacked horizontal bar chart
- Stacked vertical bar chart
- Stacked area chart
- Geolocation chart

Advanced Charts

- Geographical Heatmaps
- Key Metric Charts
- Bubble Charts
- Scatter Charts
- Progress Bar Charts



Note: Advanced Charts have their own data requirements, which may be more than the requirements for the common charts. For example, the geographical heatmaps need the latitude and longitude co-ordinates to display geographical data.

Geographical Heatmaps

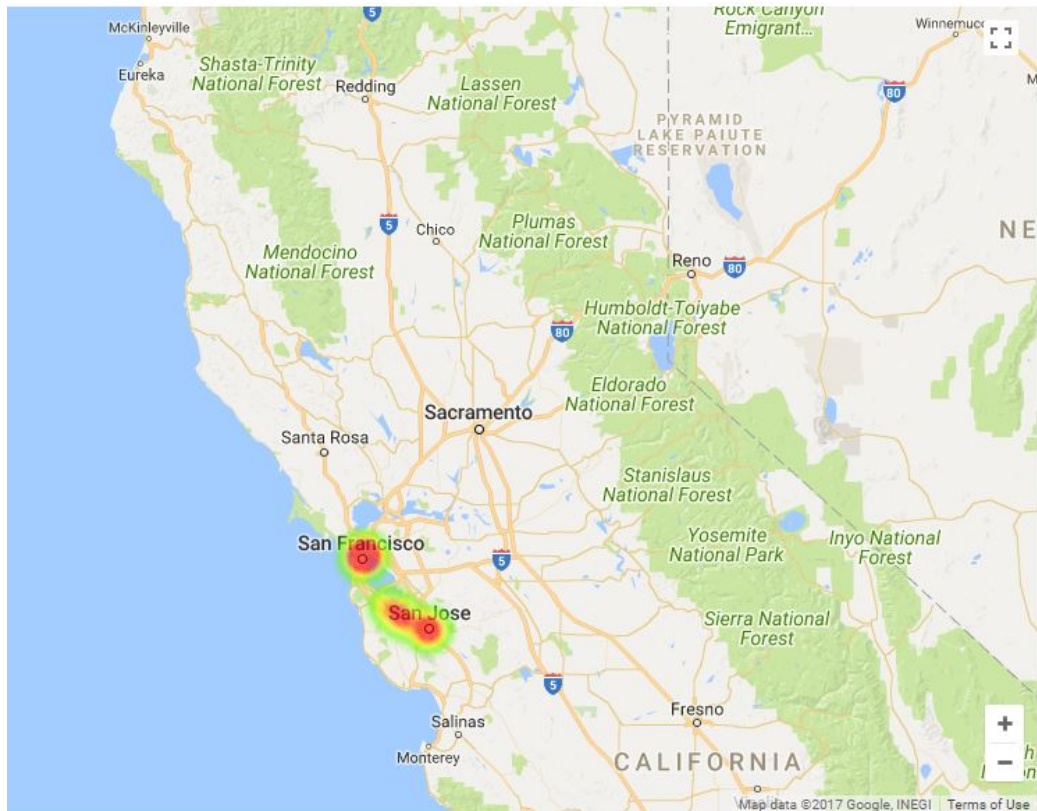
Geographical Heatmaps help you identify where the biggest concentrations of an item are, or where there are voids. Geographical Heatmaps chart type requires the user to input the following parameters:

- One data field of GeoPoint type to plot the base map. Fields of type GeoPoint accept latitude-longitude pairs to represent a physical location.
- One data field of numeric type to produce the heat map overlay.



Note: Numeric field is used only for generating Geographical Heatmaps chart type.

If there are multiple GeoPoint fields, then CA Jarvis Dashboard uses the first of those fields for painting the heat bubbles. The numeric field is used to generate the temperature for the heat bubble. We use Google Maps internal algorithms for generating heat map overlays. Here is a sample Geographical Heatmap:



A sample Geographical Heatmap

Key Metric Chart

Key Metric Chart renders only one value from the result set retrieved from the query response from CA Jarvis. The result set itself may have multiple records. CA Jarvis Dashboard picks up the top line from the query and renders the chosen field as its value.

Key Metric Chart requires you to input the following parameters:

- One data field of numeric type to render the key metric in the chart. This field is selected from a selection of numeric fields available in the result set.
- A string value to be rendered as “title”.
- A string value to be rendered as “subtitle”
- A string value to be rendered as “units”

Scatter Chart

Scatter chart renders data points on a Cartesian coordinate system where each axis has only one numeric field that is ascribed to it. Scatter chart requires you to input the following parameters:

- One data field of numeric type to render as the X Axis for the scatter chart.
- One data field of numeric type to render as the Y Axis for the scatter chart.

Bubble Chart

A Bubble Chart type is a variation on the scatter chart. The bubble chart contains a third numeric value which is used as the radius parameter for a circle that is rendered with the center at the point that is plotted on a scatter chart. Bubble Chart requires you to select the following parameters:

- One data field of numeric type to render as the X Axis for the scatter chart.
- One data field of numeric type to render as the Y Axis for the scatter chart.
- One data field of numeric type to render as the radius of the bubble that is rendered at the (x, y) position

Outlier Charts

Outlier Charts help you identify where the outliers (extreme deviation from the mean) exist in a data set. Outlier chart requires you to input the following parameters:

- The field that will be rendered on the Y Axis of the outlier chart.
- The field where the outlier criterion will be defined.
- Rules that depict the outlier points on the chart.

Cluster Charts

Cluster Charts help you understand the internal grouping structure in a data set. Cluster chart requires you to input the following parameters:

- The field that will be rendered on the Y Axis of the cluster chart.
- The field where the cluster criterion will be defined.
- Rules that depict the cluster points on the chart.

Progress Bar Charts

Progress Bar Charts help you understand the progress that is made in a process. Progress Bar chart requires you to input the following parameters:

- The numeric fields to be displayed in the chart.
- The font color to be used in the chart.
- The background color of the chart.
- Rules to choose and apply colors to the data points that befit the rule

Heat Table

Heat Tables help you understand the distribution among the data set. Heat Table requires you to input the following parameters:

- Field to be plotted in the X Axis of the heat table.

- Field to be plotted in the Y Axis of the heat table.
- Field to be plotted in the Z Axis of the heat table.
- Threshold values for the color scheme that is used in the heat table.

Box and Whiskers Chart

Box and Whiskers Charts help you understand the median, first or third percentile, and outlier in the data set. Box and Whiskers Chart requires you to input the following parameters:

- Field to be plotted in the X Axis of the heat table.
- Field to be plotted in the Y Axis of the heat table.

User Interface

- [Dashboards](#)
- [Viewlets](#)
- [Set Masthead of CA Jarvis Dashboard](#)

When you log in to CA Jarvis Dashboard, the left navigation bar of the main page has the following links:

- Dashboards
- Viewlets
- Masthead (Visible only for admin users)

These links help you navigate to Dashboards or Viewlets.

Dashboards

The dashboard default view consists of the following categories:

- All: Displays the dashboards that the user has access to.
- Private: Displays the dashboards that are created by the user.
- Default: Displays the dashboards that are published by administrative users.
- Favorite: Displays the dashboards that the user has set as favorite.

Viewlets

Following categories are part of Viewlets:

- All: Displays the viewlets that the user has access to.
- Private: Displays the viewlets that are created by the user.
- Shared with me: Displays the viewlets that are shared with the user by other users.

- Favorite: Displays the viewlets that the user has set as favorite.

Use the drop-down named Sort By to sort the objects. Click the down-arrow over your user name to change the theme of the user interface or to sign out of CA Jarvis Dashboard.



Note: If the system is configured for Ingress API Gateway, the Sign Out option may not be available.

Set Masthead of CA Jarvis Dashboard



This procedure applies only for the admin users.

Masthead is the title that appears on the top-left of the main screen of CA Jarvis Dashboard. The following procedure helps you change the Masthead. For example, if you would like to have your company name as the Masthead, use this procedure.

1. Select Masthead from the left navigation bar of the main page.
The Set Masthead screen appears.
2. Enter new value for the Masthead as a heading element in the Product Level Masthead or Tenant Level Masthead field, and click Submit.



Product Level Masthead: Change will be visible to all Product Admins.

Tenant Level Masthead: Change will be visible to all Tenant Admins.

Product Admins can set both Product Level Masthead and Tenant Level Masthead

Examples:

```
<h1> New Masthead Title </h1>
<h2> New Masthead Title </h2>
<h3> New Masthead Title </h3>
```

You can control the font size by changing the value of the <h> tag.

To revert to the original Masthead, clear the values in the Product Level Masthead or Tenant Level Masthead fields and click Submit.

Using Viewlets

- [Create a Viewlet](#)
- [Manage Viewlets](#)

Create a Viewlet

The Viewlet wizard helps you create your own Viewlets. The Viewlets operate on data that is persisted in the underlying CA Jarvis platform. Jarvis Query Language (JQL) is the query language that is provided by CA Jarvis to access data from it. JQL interface is implemented as a web-service. So, the syntax is based on the HTTP URL syntax.

You can do the following tasks using the Viewlet wizard:

- Use the wizard to build a valid JQL query.
- Manage aspects of the design of a Viewlet that are not controlled by JQL. For example, type of the chart, what should be the principal axis, and so on.

The following procedure helps you create a Viewlet.

1. Click Viewlets from the left navigation panel of CA Jarvis Dashboard.
2. Click Create a new viewlet.
3. **(Optional)** Enter Viewlet Name. If left blank, the system automatically provides a name for a Viewlet. The Viewlet name is *not* unique. You can create multiple Viewlets of the same name.
4. **(Optional)** Enter Viewlet Description. This field is searchable, with the name of the Viewlet.
5. Click **Next** to build your own JQL using the Visual Query Builder.

The system assumes that you want to use the Visual Query Builder to form the query, where the user is asked a series of questions to build a valid JQL.

6. Select a Jstream from the left pane, and click Next. Preview of the selected Jstream appears in the right pane.

A Jstream is the Jarvis equivalent of a table in RDBMS. This screen lets you select a Jstream from a list of Jstreams present in CA Jarvis. The rest of the wizard deals with the Jstream selected here.



Note: The JQL query string keeps evolving as you interact with the wizard.

7. Select an Aggregate, and click Next. An Aggregate is the Jarvis equivalent of a group-by clause in SQL. In JQL, there are several group-by clauses. This screen lets the user create group-by clauses visually.

JQL honors only the first four group-by clauses. The Viewlet wizard does not let the user create more than four group-by clauses.

The order of the clauses is also important. In this panel of the wizard, you can reorder the group-by clauses by dragging the drag icon, which becomes visible once a group by clause is applied by the user.

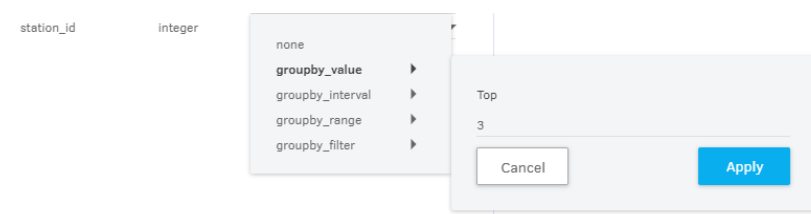
The following group-by clauses are supported:

- a. `groupby_value`: This is the most generic of group by clauses in JQL as it can be applied to all data types. No further parameters are required for this clause.
- b. `groupby_filter`: This aggregation allows the user to create different filter clauses (analogous to a where clause in SQL), which are then used as criteria for grouping by Jarvis. The system prompts the user to create filters using the provided UI.
- c. `groupby_interval`: This clause is only applicable to numeric fields. The user can provide a numeric value to create groups differentiated by the interval.
- d. `groupby_range`: This clause is only applicable to numeric fields. The user can create numerous numerical ranges (optionally with their own names) (tuples of from and to values), which can be used as criteria for grouping by the system.
- e. `groupby_georange`: This clause is only applicable to `geo_point` fields. The system expects the user to provide the following parameters:
 - i. Latitude
 - ii. Longitude
 - iii. Unit (chosen from a list of values)
 - iv. Numerous Named ranges, containing the following information
 - v. Name (optional)
 - vi. From
 - vii. To

These values permit the user to create concentric circles of ranges around the latitude and longitude that is supplied to the group by clause.

Example:

In the following image, the top three records are grouped by the Query Builder:



8. Select a Metric function, and click Next.

JQL permits computation of certain statistical metrics on data using a construct called metric functions. A metric function can be applied in the following ways:

- a. On a field where group-by is applied. Syntactically, if the metric function is applied on a query where group by clauses exist, the metric is applied to the last group by function in the query URL. Therefore, the wizard provides a re-ordering feature for the group by fields in the previous panel.
- b. Directly to a jStream.

The system permits the following metric functions:

- a.
 - i. min: Finds the minimum value in a record set.
 - ii. max: Finds the maximum value in a record set.
 - iii. avg: Sums up all values and then calculates the average.
 - iv. sum: Sums up all values
 - v. count: Returns the number of rows that matches a specified criteria. Count is the only metric that can be applied to non-numeric fields.
 - vi. stats
 - vii. ext_stats

9. Create a Filter, and click Next.

A Filter is the Jarvis equivalent of a where clause in SQL. JQL queries, whether they use aggregations or simple select, permits a logical filtration of the result set.

A filter is a logical concatenation of basic filter expressions. The Viewlet wizard uses the following definition for a filter expression:

`filter_expression = <fieldname> binary_operation <value>`

Multiple filter expressions can be concatenated using parentheses and logical operands of AND and OR. For example,

`(fieldname1 > 4).AND.(fieldname2 < 10)`

The filter mechanism of the Viewlet wizard does not permit an arbitrary expression concatenation. It allows you to create many expression types as follows:

- All filter expressions that are associated with disparate fields are always ANDed with each other.
- All filter expressions that are associated with a given field can either be ANDed or ORed among each other.

10. Select Other JQL parameters, and click Next.

This panel renders different values when reached through different routes:

- a. If a user creates a simple select-type query, the system asks for the following values:
 - i. Top: this number tells CA Jarvis to send the first “top” number of records only.
 - ii. Skip: this number tells CA Jarvis to skip the first “skip” number of records and then send the next “top” number of records.
- b. If the user makes an aggregate or metric query, the system only asks for the value of the Top variable.

11. Select the Column Name, sort order, and click Next.

After querying CA Jarvis, the system renders the following bits of functionality:

- a. List of fields that are returned from the query: User gets to see a list of fields with checkboxes in a grid. By checking and unchecking the checkboxes, the user selects a subset of the original list to be rendered in a chart.
- b. Change field order: User can drag and change the order of fields.
- c. Change field name: User can edit the display name of the field by simply overwriting the text in the second column.
- d. Add another field: This button lets the user add a custom “calculated field” (explained later in this document) to the grid.

12. Select a Chart Type of your choice, and click Next.

You can change the chart type that is selected here later by using the Viewlet edit screen. You can select values like Principal Axis, Auto-refresh in, Split Dimension, and so on, if you want:

Principal Axis: The term principal axis is not precise, but conveys the idea intuitively. In many chart types, such as Line, Area, and Bar chart types, users generally want to plot one field on the X-Axis and the rest of the fields on the Y-Axis – we will call X-Axis as the Principal Axis, as this general understanding is rapidly thwarted by horizontal bar charts, where the axes undergo a role reversal.

User’s choosing a principal axis informs the system to use a particular column as the preferred axis as against all other columns, which would be plotted on the other axis. The user can not to select a principal axis. In that event, the system provides an integral series (1, 2, 3, ...) as the principal axis and plots the rest of the fields on the other axis.

Auto-refresh in: The system permits a configurable value to be associated to a viewlet whereby the viewlet refreshes itself after every few n seconds when rendered in a dashboard view. This field has a drop-down list of values (including never) that can be applied to a viewlet. This value cannot be changed at the runtime.

Split Dimension: If the user wants to create a pivot table view for a given dimension, then user can select a dimension to split in this field.

Viewlet X-axis Label and Viewlet Y-axis Label : The labels to be used for X axis and Y axis while plotting the chart.

13. Select Value Fields and click Create. Select these fields, only if you have provided value in the Split Dimension field to create a pivot table view.

Here is the sequence of events that happen when you click the Create button:

- a. Saves the Viewlet in the CA Jarvis Dashboard persistence layer.
- b. Navigates you to the viewlet editing screen.
- c. Renders the Viewlet in the chart-type that is chosen in the Viewlet creation wizard
- d. (Applies only for Advanced Charts) Displays the following screen when you select the advanced charts. Select the chart type, input required values, and click Apply:

Name of the Chart	Input
Geographical Heatmaps	One data field of GeoPoint type to plot the base map. Fields of type GeoPoint accept latitude-longitude pairs to represent a physical location.

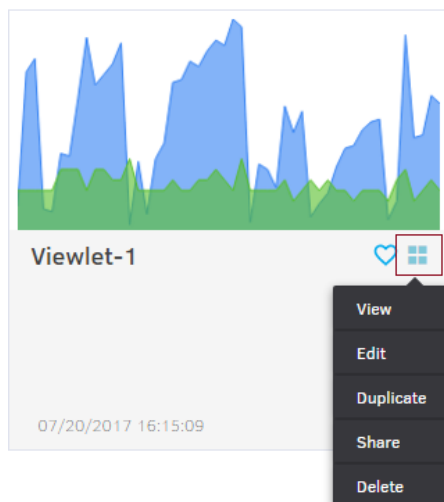
Name of the Chart	Input
Key Metric Chart	<p>One data field of numeric type to render the key metric in the chart. This field is selected from a selection of numeric fields available in the result set.</p> <p>Title: A string value to be rendered as title in the chart. Subtitle: A string value to be rendered as subtitle in the chart. Units: A string value to be rendered as units in the chart.</p>
Bubble Chart	<p>X-Axis: One data field of numeric type to render as the X Axis for the scatter chart. Y-Axis: One data field of numeric type to render as the Y Axis for the scatter chart. Bubble radius: One data field of numeric type to render as the radius of the bubble that is rendered at the (x,y) position</p>
Scatter Chart	<p>X-Axis: One data field of numeric type to render as the X Axis for the scatter chart. Y-Axis: One data field of numeric type to render as the Y Axis for the scatter chart.</p>
Outlier Charts	<p>Data Field: The field that will be rendered on the Y Axis of the outlier chart. Outlier Indicator: The field where the outlier criterion will be defined. Outlier Criterion: Rules that depict the outlier points on the chart.</p>
Cluster Charts	<p>Data Field: The field that will be rendered on the Y Axis of the cluster chart. Cluster Indicator: The field where the cluster criterion will be defined. Cluster Criteria: Rules that depict the cluster points on the chart.</p>
Progress Bar Charts	<p>Value field: The numeric fields to be displayed in the chart. Text color: The font color to be used in the chart. Background: The background color of the chart. Progress Bar Coloring rules: Rules to choose and apply colors to the data points that befit the rule. Use the following steps to define the rule:</p> <ol style="list-style-type: none"> Select a binary operator from the dropdown {=, !=, <, >, <=, >=} Insert a numerical value in the edit box Pick a color. <p>The system applies the rules one at a time. The rules are applied from top to bottom as they appear in the Edit screen. Latter rules override the colors applied by the former rules.</p>
Heat Table	<p>X Axis: Field to be plotted in the X Axis of the heat table. Y Axis: Field to be plotted in the Y Axis of the heat table. Z Axis: Field to be plotted in the Z Axis of the heat table. Start Color and End Color: Threshold values for the color scheme that is used in the heat table.</p>
Box and Whiskers Chart	<p>X Axis: Field to be plotted in the X Axis of the Box and Whiskers chart. Y Axis: Field to be plotted in the Y Axis of the Box and Whiskers chart.</p>

Manage Viewlets

You can navigate to the Viewlet editing screen using one of the following ways:

- At the end of the Viewlet creation wizard.
- From the Viewlet Browse Screen by clicking the Edit menu from the Viewlet Icon.
- From Viewlet list in the dashboard editing screen.
- By directly editing from the Viewlet panel in the dashboard.

The icon that is highlighted in the following image provides the options to manage your Viewlets. The options may vary based on your permissions over the Viewlet:



The following points summarize how you can manage Viewlets:

- **View:** You can view a Viewlet using the View option or by directly clicking the Viewlet. In View mode, you can do the following task:
 - Edit the Viewlet. Appears only if you have the permission to edit the Viewlet.
 - Share the Viewlet.
- **Edit:** You can edit a Viewlet using the Edit option. In Edit mode, you can do the following task depending upon your permissions:
 - Duplicate the Viewlet.
 - Share the Viewlet.
 - Delete the Viewlet.
 - Change name and description of the Viewlet.
 - Modify the type of chart that is used in the Viewlet. Select the required type of chart under the Chart/Advanced Chart label and click Save. Certain constraints exist on which a chart type can be created from the given data type combination of the fields in the resultset. For example, for creating a Geolocation type chart, one of the fields must be a geo_point, a CA Jarvis data-type that only accepts lat/long tuple values.

- Edit the Viewlet using the Edit this viewlet icon.
- Brush and Zoom (Brush-bar). This feature helps you perform scale and shift operations to the data being viewed.

Shifting is the process of changing the location of the graph without changing the size or shape of the graph. Scaling is the process of changing the size and shape of the graph. The feature is available to the following viewlet types:

- Bar Chart
- Stacked Bar Chart
- Line Chart
- Area Chart
- Stacked area chart

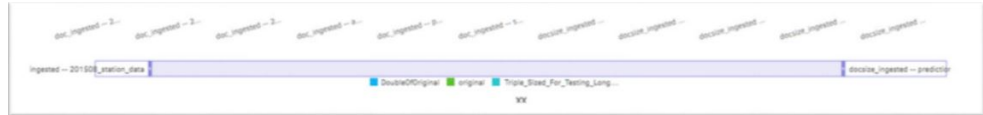
The following screenshot shows the Brush and Zoom feature available in the Viewlet edit screen:



When the Use Brush-Bar option is toggled on, a bar similar to the following screenshot appears below the Viewlet:



You can zoom into the data by moving either end of the bar. You can shift the bar in either direction to shift the data shown in the visualization. Scale and shift operations respect the natural boundary conditions of the dataset. Both operations are performed on the data fetched and retained in the browser. These operations do not re-query data from CA Jarvis.



toggled toggled

- **Duplicate:** You can duplicate a Viewlet using the Duplicate option. The system produces a duplicate Viewlet and stores it with an alternative name. If the Viewlet being duplicated was received by a user through sharing, then the freshly created Viewlet is no longer a shared Viewlet. So using duplication, you can change the ownership of a Viewlet object.
- **Share:** You can share Viewlets with other users of CA Jarvis Dashboard individually or in a batch mode. After selecting the Viewlets to be shared, enter Usernames of the users with whom you want to share the Viewlets with, and click Send. Separate the user names with commas.

Search Viewlet

Sort By: Date modified (new to old)

Select All

Clear All

☐ Outlier charts - Baseline
☐ Minimum Humidity - Progress...
☐ JMetrics - user_created
☐ JMetrics - dashboard_views
☐ JMetrics - viewlet_views
☐ JMetrics - viewlet_created
☐ JMetrics - dashboard_created
☐ JMetrics - Baseline
☐ Harry
☐ Dick

Share Viewlets with others

Enter Usernames of the users you want to share these viewlets with.

Separate them with commas.

Cancel

Send

This list of viewlets has been shared with the following users.

Close

- **Delete:** You can permanently delete a Viewlet using the Delete option. You can delete a Viewlet using one of the following ways:
 - From the drop-down list
 - From the viewlet edit screen
 - From the dashboard edit screen

Using Dashboards

- [Create a Dashboard](#)
- [Manage Dashboards](#)
 - [Viewing Timeline Comparison Dashboard](#)
 - [Sharing of Dashboards](#)

Create a Dashboard

The Dashboard Creation wizard helps you create your own Dashboards.

The following procedure helps you create a Dashboard.

1. Click Dashboards from the left navigation panel of CA Jarvis Dashboard.
2. Click Create a new dashboard.
3. Enter Dashboard Name.
The Dashboard name is not unique. You can create multiple Dashboards with the same name.
4. **(Optional)** Enter Dashboard Description.
This field is searchable with the name of the Dashboard.
5. Enter the Default Days for Time Filter.
This value is used by the time filter as the default number of days used by Automatic Time-Filter in the dashboard view screen of the system. At the time, of the installation, each tenant can be prescribed a numeric value, which is then copied in this field by the system. You can overwrite the value in this screen.
6. **(Optional)** Enable TimeLine Comparison. If you enable this option, caption of the previous field changes to Default Period, and you can specify how the comparison should happen. For example, you can compare data from previous day against today's data. There is no way of reversing a time-series dashboard to a regular dashboard and vice-versa once created.
7. Click Next.
8. Select a template for the Dashboard, and click Create.
The system creates an empty Dashboard.
9. Drag-and-drop the Viewlets that are displayed in the left navigation panel to the blank panels of the Dashboard.
Viewlets can be dragged from panel to panel too. You can add the same Viewlet to more than one Dashboard.

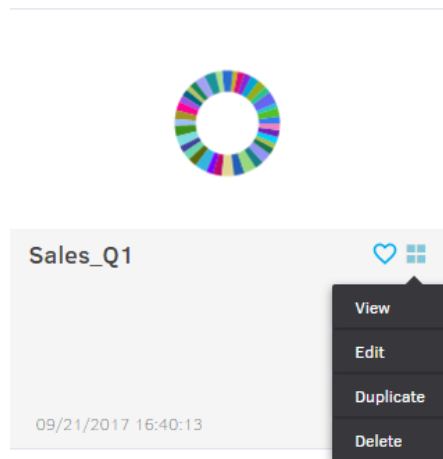


Only those Viewlets that satisfy one of the following conditions can be used in a timeline comparison dashboard:

- Datetime field is used in the principal axis of the Viewlet.
- Groupby time interval clause is used in the JQL with the following values ('day', 'hour', '30m').
- Pivot tables are not used.

Manage Dashboards

To view all the Dashboards that you have the permission to view, click Dashboards from the left navigation bar of the main page of CA Jarvis Dashboard. The icon that is highlighted in the following image provides the options to manage your Dashboards. The options may vary based on your permissions over the Dashboard:



The following points summarize how you can manage Viewlets:

- **View:** You can view a Dashboard using the View option or by directly clicking the Dashboard. In View mode, you can do the following tasks:
 - Apply filters to refine the data that is displayed in the Viewlets.
 - Print the Dashboard.
 - Edit the Dashboard. Appears only if you have the permission to edit the Dashboard.
- **Edit:** You can edit a Dashboard using the Edit option. In Edit mode, you can do the following tasks depending upon your permissions:
 - Duplicate the Dashboard.
 - Delete the Dashboard.
 - Change name and description of the Dashboard.
 - Edit Viewlets from the Dashboard. Do the following steps to edit a Viewlet from the Dashboard:
 1. Hover over a Viewlet.
 2. Click the icon that appears on the lower-left corner of the Viewlet.
After the changes are saved, you can return to the Dashboard Edit screen by clicking the Dashboard link on the top left of the screen.



Note: You can also edit a Viewlet from the left-side pane where the Viewlets are listed.

- Delete a Viewlet from the Dashboard. A Viewlet can be removed from a dashboard in the edit mode only. To remove a Viewlet from the dashboard, follow these steps in edit mode:

1. Hover over a Viewlet.
2. Click the icon that appears on the lower-right corner of the Viewlet. This step only removes the Viewlet from the Dashboard. The original Viewlet does not get deleted.

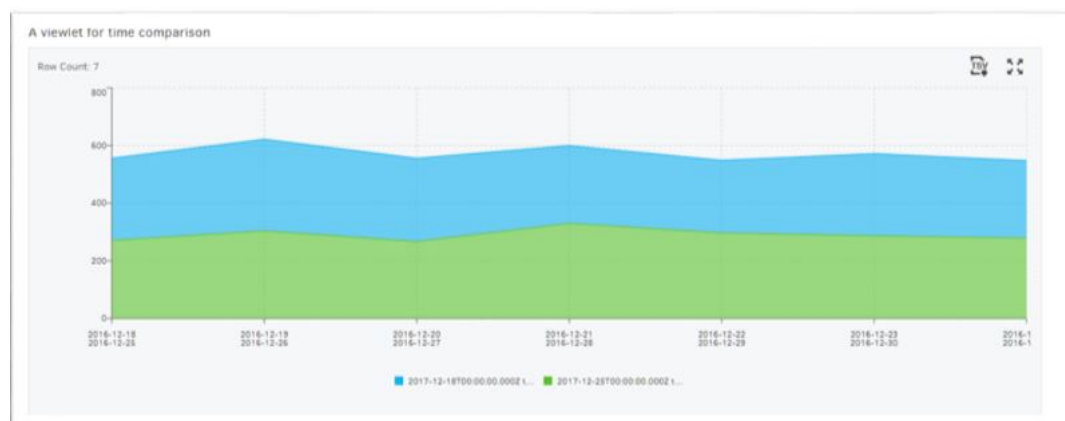


Note: You can also delete a Viewlet using the delete icon from the left-side pane where the Viewlets are listed. This option permanently deletes the Viewlet.

- Drag-and-drop the Viewlets that are displayed in the left navigation panel to the blank panels of the Dashboard
- **Duplicate:** You can duplicate a Dashboard using the Duplicate option. The system produces a duplicate object and stores it with an alternative name. If the object being duplicated was received by a user through sharing, then the freshly created Dashboard is no longer a shared Dashboard. So using duplication, you can change the ownership of a Dashboard object.
- **Delete:** You can permanently delete a Dashboard using the Delete option.

Viewing Timeline Comparison Dashboard

In timeline comparison mode, every visualization is of timeline type. The timeline graphs from two separate time windows are superimposed on the principal axis of same canvas. Here is an example:



As seen in the example, the principal axis has two time windows superimposed on each other.

The automatic time filter is different for this type of dashboard. The timeline uses a pre-specified (Default) set of time windows. There is also an option for defining customized time (Custom) windows for period comparison.

Type: **Default**

Compare: **Custom** Week v Last Week ▼

Period1: 2017-12-25 00:00:00 to 2017-12-31 23:59:59

Period2: 2017-12-18 00:00:00 to 2017-12-24 23:59:59

Cancel **Apply**

In **Default** Mode the following period templates are available:

- Today v yesterday
- This week v last week
- This month v Last Month
- This Quarter v Last Quarter

In the **Custom** Mode, do the following:

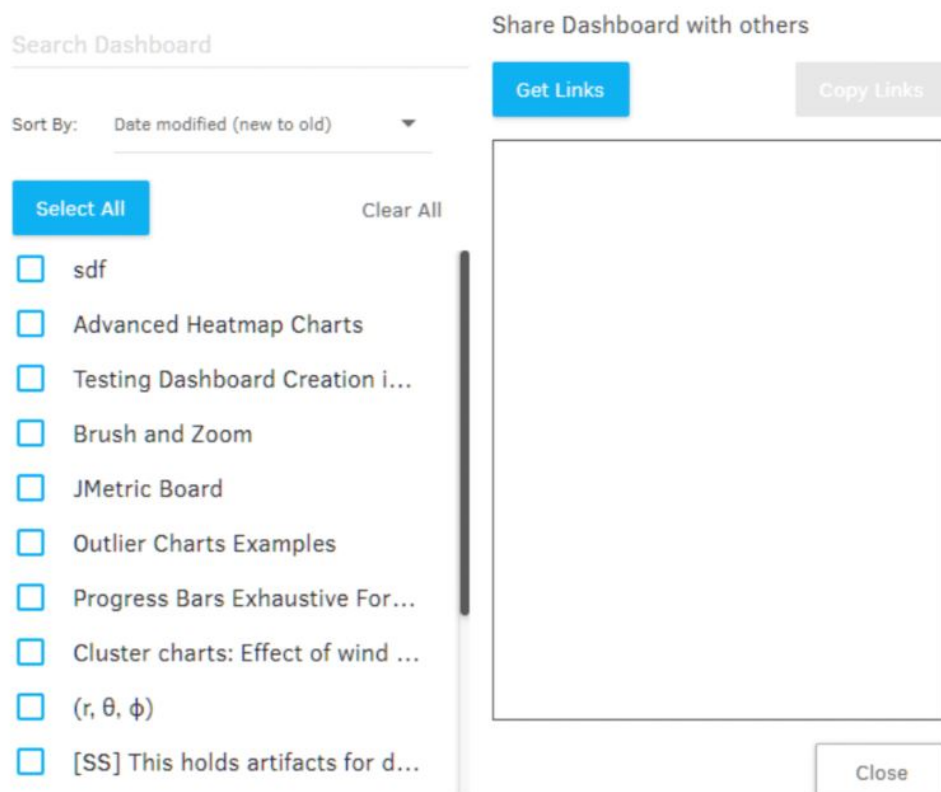
1. Choose one of the following period units:
 - a. Period
 - b. Week
 - c. Month
2. customize the following independent parameters
 - a. First Point
 - b. Compare Point

After selecting the filters, the system automatically performs the timeline comparison on all Viewlets in the dashboard.

Sharing of Dashboards

You can share one or more Dashboards (the URL of a Dashboard) to other users of the same tenant. The following procedure explains how you can share Dashboards:

1. Click the Share link on the dashboard browse page to share the Dashboards.
A screen similar to the following appears:



2. Select the Dashboards to be shared.
3. Click Get Links to create unique URLs for your selections.
The generated links appear on the list box.
4. Click Copy Links to copy the links to your clipboard.
You are ready to share the links electronically (through email, messaging apps, and so on) with the users.

API Reference of CA Jarvis Dashboard

- [Product Onboarding API](#)
 - [Prerequisites](#)
 - [Onboard a Product](#)
 - [Update a Product](#)
 - [Delete a Product](#)
- [Tenant Onboarding API](#)
 - [Prerequisites](#)
 - [Onboard A Tenant](#)
 - [Update a Tenant](#)
 - [Delete a Tenant](#)
- [User Onboarding API](#)
 - [Prerequisites](#)
 - [Onboard A User](#)

- [Update a User](#)
- [Delete a User](#)
- [List User Name](#)
- [Default Dashboard Onboarding](#)

After installing the CA Jarvis Dashboard, use the following APIs to onboard products and tenants.



You must onboard your product before onboarding the tenant.

- Product onboarding API
- Tenant onboarding API

SSL	Non-SSL
8443 and 8080 are the default ports. If necessary, you can change the port in the lddsin installer properties file.	
8443 as the <port number>.	8080 as the <port number>.
https as the scheme in the URL.	http as the scheme in the URL.

Product Onboarding API

Use the Product Onboarding API to create a product.

Prerequisites

- A running instance of CA Jarvis should be accessible from CA Jarvis Dashboard.
- The productCode that you are going to onboard in the CA Jarvis Dashboard should exist in the connected instance of CA Jarvis (product_id).

Onboard a Product

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/product/`

A sample payload follows:

```
POST LDDS-rest/onboarding/product HTTP/1.1
Host: <example.org>
Content-Type: application/json

BODY
{
  "productCode": "babs",
  "jarvisUrl": "http://www.jarvis.com:8888",
  "docTypeFilter": "<regex>",
  "googleMapKey": ""
}
```

Parameter	Description	Required?
productCode	Identical to product_id in Jarvis. Ensure that you provide a valid productCode. Get it from your Jarvis admin.	Yes

Parameter	Description	Required?
jarvisUrl	Fully Qualified Domain Name (FQDN) of the machine where CA Jarvis is installed. Contact your Jarvis admin for this URL.	Yes
docTypeFilter	Provide a regular expression to filter the DocTypes. All DocTypes whose name matches the regular expression provided are excluded from LDDS operations. For example, if you specify <i>apim_*</i> , all the jstreams starting with <i>apim</i> are excluded. This parameter filters only the viewlet and filter authoring system. If an imported dashboard references an Index that is specified to be filtered out, the dashboard displays the data.	Optional
googleMapKey	Google Maps require an API key to make it functional. Provide the key, if available. If not, leave it empty.	Optional

Response Codes

Code	Status
200	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

Update a Product

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/product//<productId>`

A sample payload follows:

```
PUT LDDS-rest/onboarding/product/<productId> HTTP/1.1
Host: <example.org>
Content-Type: application/json

BODY
{
  "productCode": <new_product_id>,
  "jarvisUrl": <new_jarvis_url>,
  "docTypeFilter": "<regExp>",
  "googleMapKey": <new_google_map_key>
}
```

Response Codes

Code	Status
200	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

Delete a Product

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/product/<productCode>`

A sample payload follows:

```
DELETE LDDS-rest/onboarding/product/<productCode>
Host : <example.org>
Content-Type: application/json
```

Response Codes

Code	Status
200	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

Tenant Onboarding API

Use Tenant Onboarding API to create a tenant.

Prerequisites

- A running instance of CA Jarvis should be accessible from the CA Jarvis Dashboard.
- The tenantCode that you are going to onboard in the CA Jarvis Dashboard should exist in the connected instance of CA Jarvis (tenant_id).

Onboard A Tenant

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/product/<productCode>/tenant`



The name of the product (productCode) under which the tenant should be on-boarded must be provided in the URL

A sample payload follows:

```
POST LDDS-rest/onboarding/product/<productCode>/tenant
HTTP/1.1
Host: <example.org>
Content-Type: application/json
BODY
{
  "tenantCode": "<tenantCode>",
  "productAdminRoleName": "PA",
  "adminRoleName": "TA",
  "userRoleName": "OA",
  "ana": "internal"
}
```

Parameter	Description	Required?
tenantCode	Identical tenant_id in Jarvis. Ensure that you provide a valid tenantCode. Get it from your Jarvis admin.	Yes
productAdminRoleName	Name for the user group of product admins.	Yes
adminRoleName	Name for the user group of tenant admins.	Yes
userRoleName	Name for the user group of normal users.	Yes
ana		Yes

Parameter	Description	Required?
	ana stands for Authentication and Authorization, This variable can have the following values:	
	<i>Internal</i> : To use CA Jarvis Dashboard as the identity store.	
	<i>Ingress</i> : To use APIM as the identity store	

Response Codes

Code	Status
200	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

Update a Tenant

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/product/<productId>/tenant/<tenantId>`

A sample payload follows:

```
PUT LDDS-rest/onboarding/product/<productId>/tenant/<tenantId> HTTP
/1.1
Host: <example.org>
Content-Type: application/json

BODY
{
  "tenantCode": <new_tenant_id>,
  "productAdminRoleName": <new PA role name>,
  "adminRoleName": <new TA role name>,
  "userRoleName": <new OA role name>
}
```



All the parameters are optional.

You must not update the value of *ana*.

Response Codes

Code	Status
200	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

Delete a Tenant

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/product/<productCode>/tenant/<tenantCode>`

A sample payload follows:

```
DELETE LDDS-rest/onboarding/product/<productCode>
Host : <example.org>
Content-Type: application/json
```

Response Codes

Code	Status
200	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

User Onboarding API

Use User Onboarding API to create and manage users.

Prerequisites

- A running instance of CA Jarvis should be accessible from CA Jarvis Dashboard.
- The tenantCode that you are going to onboard in CA Jarvis Dashboard should exist in the connected instance of CA Jarvis (tenant_id).

Onboard A User

This method helps administrators create users with the supplied credentials in the body of the request within the scope of a productCode/tenantCode in CA Jarvis Dashboard.

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/user`



The name of the product (productCode) under which the tenant should be on-boarded must be provided in the URL

A sample payload follows:

```
POST LDDS-rest/onboarding/user HTTP/1.1
Host: <example.org>
Content-Type: application/json
```

```
BODY
{
  "productCode": "productCode",
  "tenantCode": "tenantCode",
  "username": "user@ca.com",
  "password": "myPassword@123",
  "authGroup": "0A"
}
```

Parameter	Description	Required?
productCode	Identical to product_id in Jarvis. Ensure that you provide a valid productCode. Get it from your Jarvis admin.	Yes
tenantCode	Identical tenant_id in Jarvis. Ensure that you provide a valid tenantCode. Get it from your Jarvis admin.	Yes

Parameter	Description	Required?
username	Name for the user to be created. Provide the email address of the user.	Yes
password	Password for the user.	Yes
authGroup	Provide the group in which the user belongs to. authGroup takes one of the following values only: <ul style="list-style-type: none"> ▪ OA ▪ TA ▪ PA 	Yes

Response Codes

Code	Status
20x	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

Update a User

This method helps administrators do one or both of the following tasks:

- Change the password of a user associated with the supplied username.
- Change the authGroup of the user.

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/user`

A sample payload follows:

```
PUT LDDS-rest/onboarding/user HTTP/1.1
Host: <example.org>
Content-Type: application/json

BODY
{
  "productCode": "productCode",
  "tenantCode": "tenantCode",
  "username": "user@ca.com",
  "password": "Abc12#11",
  "authGroup": "OA"
}
```

If the user exists within the scope of the supplied productCode/tenantCode and the supplied password is not null, the service attempts to update the user record with the new password value.



You must provide either password or authGroup with this service. You can provide both password and authGroup too.

Response Codes

Code	Status
20x	Success

40x

Error

For more information about the error codes, see [Troubleshooting](#).

Delete a User

This method helps administrators delete a given user record associated with the value of productCode / tenantCode/username combination supplied with the request.

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/user`

A sample payload follows:

```
DELETE LDDS-rest/onboarding/user HTTP/1.1
Host: <example.org>
Content-Type: application/json
```

BODY

```
{
  "productCode": "productCode",
  "tenantCode": "tenantCode",
  "username": "user@ca.com"
}
```

If the productCode, tenantCode, username combination exists in the database, the service attempts to delete the user record. All related entities associated with the user (viewlets and owned and shared dashboards) are deleted.

Response Codes

Code	Status
200	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

List User Name

This method helps administrators list validate the existence of a username within the product /tenant scope in the CA Jarvis Dashboard database.

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/user/`

A sample payload follows:

```
GET LDDS-rest/onboarding/user HTTP/1.1
Host: <example.org>
HTTP Headers
username: "user@ca.com"
productCode: "productCode"
tenantCode: "tenantCode"
```



All the parameters are mandatory.

Response Codes

Code	Status
200	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

Default Dashboard Onboarding

An admin user creates default dashboards and publishes the dashboards in the admin environment. For example, you can design a sales-trend dashboard to make it available to your staff. These dashboards appear under the Default category in the left-navigation pane under Dashboards. Use the **Export** button to create a portable .JSON file inside the download folder of your browser. Use this file as an HTTP Post Form-Data in the API calls to upload the default dashboards. The uploaded default dashboards are applied to all the tenants under a given product. That is, the default dashboards appear for all users. For each upload, the system removes all existing default dashboards for that product and replaces them with the latest set of dashboards.

After Onboarding a product and any number of tenants under it, use the following API to upload the default dashboards. This API updates all tenants simultaneously.

Resource URL: `http(s)://<host name or IP address>:<port number>/LDDS-rest/onboarding/product/<productCode>/cannedDashboards`

A sample payload follows:

```
POST LDDS-rest/onboarding/product/<productCode>/cannedDashboards HTTP/1.1
Host: <example.org>
Content-Type: multipart/form-data

Content-Disposition: form-data; name="file"; filename="<dashboard>.json"
```

Replace *<dashboard>* with the name of the exported .JSON file.

Response Codes

Code	Status
200	Success
40x	Error

For more information about the error codes, see [Troubleshooting](#).

Integration

This section describes how to use CA Jarvis Application Programming Interfaces (APIs).

- [Jarvis Analytics Framework \(JAF\)](#)

Jarvis Analytics Framework (JAF)

CA Jarvis enables enterprises to build analytics solutions. Data scientists and engineers can build, track, and deploy predictive models by using Jarvis REST APIs on demand to avoid re-implementing complex algorithms or data cleaning logic. CA Jarvis accelerates data science from exploration to deployment, enabling faster return on investment on analytical activities. This section describes the following articles:

- [Getting Started with JAF SDK](#)
- [Set Up Development Environment](#)
- [How to Use JAF SDK?](#)
- [JAF API Reference](#)

Getting Started with JAF SDK

- [What is JAF SDK?](#)
- [JAF SDK Architecture](#)
- [Development Phase and Production Phase](#)
 - [Data Science Algorithm Development Phase](#)
 - [Data Science Algorithm Deployment Phase](#)

What is JAF SDK?

Jarvis Analytics Framework Software Development Kit (JAF SDK) for Python allows Data Scientist to integrate data science algorithms with CA Jarvis. You can use the JAF SDK for Python to write Python applications that interact with CA Jarvis.

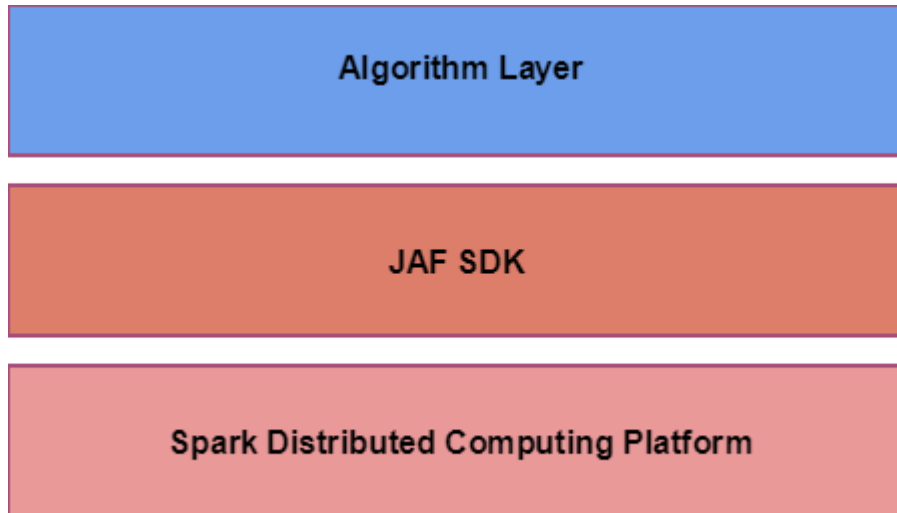
Data Scientists can develop applications to generate insights from historical data that is stored in HDFS (batch job), or from streaming data that is queued in Kafka (speed job).

JAF SDK uses Apache Spark as the distributed (cluster) computing framework, and it exposes Apache Spark data structures for data processing.

JAF SDK provides custom interfaces for interacting (read/write) with CA Jarvis. For any data computing operations, Data Scientist can use Apache Spark supported libraries and computing paradigm.

JAF SDK Architecture

The following diagram illustrates the architecture of JAF SDK:



Component	Description
JAF SDK	JAF SDK helps Data Scientist integrate data science algorithms with CA Jarvis. The layer provides various APIs for executing both batch and speed jobs. For example, API to read data from HDFS (Storage) and to expose data as Spark Dataframe.
Algorithm Layer	<p>JAF SDK can read and store data either in the Speed or Batch layer. The Algorithm layer contains the logic for processing the data. Any custom configuration that is required to the layer is made available as Spark Broadcast variable. Broadcast variables allow you to keep a read-only variable cached on each server rather than shipping a copy of it with the tasks.</p> <p>Batch Job works as a Spark Batch Job utilizing all underneath resources and JAF for data access and cache operations. In the batch processing model, the analytical job runs on historic data batch wise. For example, the job runs on data for the last two weeks or on data for the last 6 months.</p> <p>Speed Job works as a Spark Streaming application utilizing all underneath resources and JAF for data access and cache operations. In the stream processing model, near-real-time data is directly fed into an analytics system, for example, CA Jarvis, for processing.</p>
Spark Distributed Computing Platform	This layer exposes the distributed computing paradigm.

Development Phase and Production Phase

You must understand the following phases before developing your data science algorithms:

- Data Science Algorithm Development Phase
- Data Science Algorithm Deployment Phase

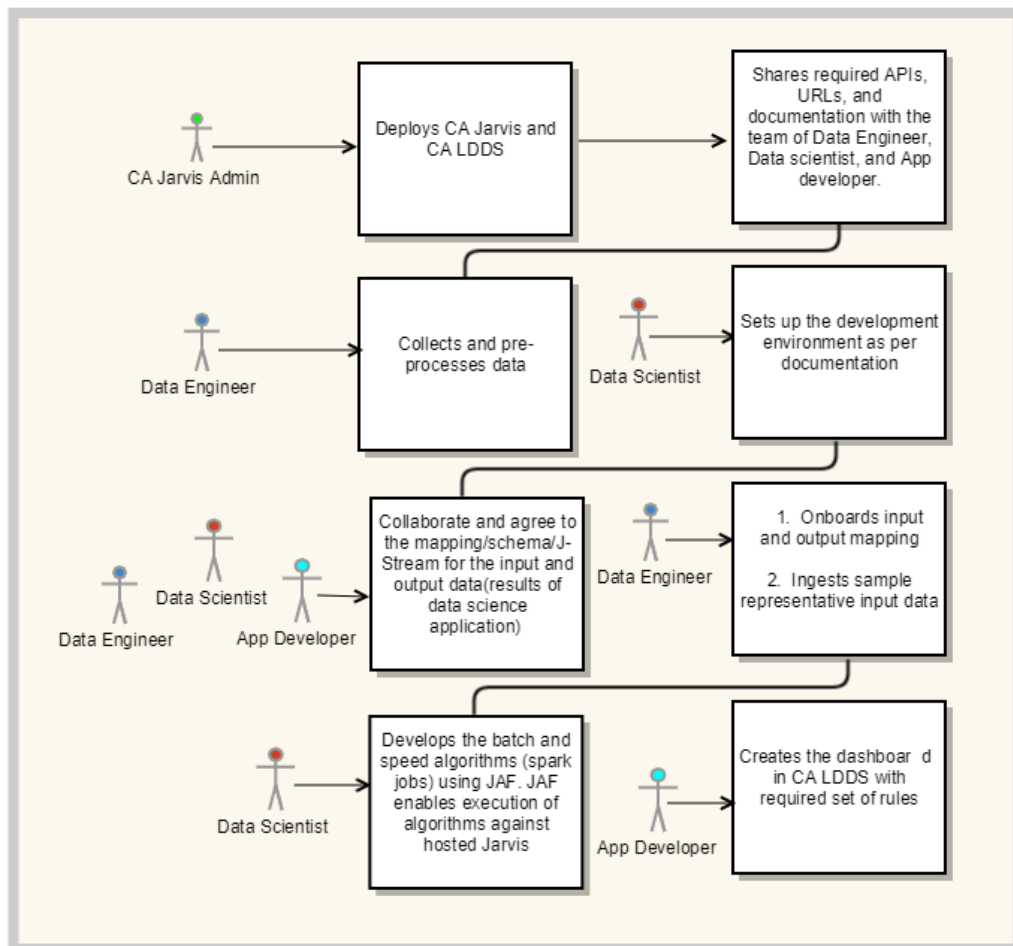
Data Science Algorithm Development Phase

In this phase, the CA Jarvis platform is hosted on a local server, and Data Scientists can develop data science applications (batch and speed) on their local computer. The Data Science Algorithm Deployment Phase ensures sufficient integration testing. The following steps describe the workflow that happens during the development phase:



Note: In this phase developers are using local (development server) Spark, so use representative data. In the deployment phase, Spark cluster on Production Jarvis setup is used as the distributed computing engine.

Development Phase



Data Science Algorithm Deployment Phase

Once sufficient integration testing is done at development phase, use Jarvis on-boarding APIs to on-board product, tenant, input doc_types, output doc_types, and the newly developed algorithm into the Jarvis production system. In this phase, the developed application is deployed and run on Jarvis in the distributed computing mode. The following steps describe the workflow that happens during the deployment phase:

1. [Use Jarvis API to On-board Product](#)
2. [Use Jarvis API to On-board Tenant](#)
3. [Use Jarvis API to On-board input and output docTypes](#)
4. Use Jarvis API to On-board Data Science applications for docType
 - a. [For Speed jobs](#)
 - b. [For Batch jobs](#)
5. [Use the CA Jarvis Dashboard to visualize results.](#)



If a running speed job fails due to any reason, currently we resubmit the job. In production environment, we highly recommend HA setup to avoid the scenario.

Set Up Development Environment

Ensure that your development setup is ready with the following prerequisites:

Supported Platforms	<ul style="list-style-type: none"> ▪ MAC: macOS Sierra or macOS El Capitan ▪ Windows: Windows 10 or Window 8.1
Software requirements:	<ul style="list-style-type: none"> ▪ Spark 2.2 ▪ Python 2.7.x with pip configured (Python 3 is not supported) ▪ Pycharm (Suggested IDE) ▪ JDK 8
Environment variables	<p>Windows:</p> <ul style="list-style-type: none"> ▪ SPARK_HOME: Set the full path of your spark folder as the value. ▪ PYTHONPATH: Set %SPARK_HOME%\python as the value. <p>MAC:</p> <p>Edit <i>bash_profile</i> to append the following values. Ensure that you edit the value for full path of your spark folder:</p> <ul style="list-style-type: none"> ▪ export SPARK_HOME= Set the full path of your spark folder as the value. ▪ export PYTHONPATH=\$SPARK_HOME/python

1. Obtain and unzip the latest version of JAF SDK package.
2. Open Command Prompt, and change to the `\jarvis_byods` folder.
3. Run the following command to install the Python prerequisite packages for JAF SDK:


```
sudo pip install -r requirements.txt
```

4. Use the outline python code provided inside the *custom_app* folder to develop your own speed or batch application.

5. Onboard the development phase product, tenant and required doc types:

- a. [Use Jarvis API to On-board Product](#)
- b. [Use Jarvis API to On-board Tenant](#)
- c. [Use Jarvis API to On-board input and output docTypes](#)

6. Retain the folder structure provide with the outline project.

- a. If your own application requires additional resources other than the Python prerequisite packages that are installed in step 3, specify those in "additional_requirements.txt" and install them using the following command. Note that this file will be used during packaging your app before onboarding:

```
pip install -r additional_requirements.txt
```

- b. Place any application-specific configuration files that need to be used as cache (spark broadcast variable) in the "config" folder while configuring the files.

- c. Contact Jarvis admin and obtain values needed to be provided in the following files located in the *jarvis_byods\custom_app* folder:

- i. **jarvis_byods_config_batch.json**: Templates for batch application. Contact Jarvis Admin for the values. Provide the development phase product, tenant, and required doc types.

```
{
  "hdfs_url": "hdfs://<jarvis_host>:9000",
  "kafka_brokers": "<jarvis_host>:9092",
  "out_topic": "CAA_unverified",
  "product_id": "<product>",
  "job_id": "<uniqueId_per_product>",
  "in_doctype_list": [
    {
      "doc_type_id": "<input_data_doctype>",
      "doc_type_version": "<input_data_doctype_version>"
    }
  ]
}
```

- ii. **jarvis_byods_config_speed.json**: Templates for speed application Contact Jarvis Admin for the values. Provide the development phase product, tenant, and required doc types.

```
{
  "microbatch_interval": 30,
  "hdfs_url": "hdfs://<jarvis_host>:9000",
  "schema_registry_url": "http://<jarvis_host>:8082",
  "kafka_brokers": "<jarvis_host>:9092",
  "out_topic": "CAA_unverified",
  "product_id": "<product>",
  "job_id": "<uniqueId_per_product>",
  "batch_job_id": "<model_producer_id_optional>",
  "in_doctype_list": [
    {
      "doc_type_id": "<input_data_doctype>",
      "doc_type_version": "<input_data_doctype_version>",
      "kafka_topic_list": [
        ["CAA_DTR_SV_BL_<product>_<input_data_doctype>_<input_data_doctype_version>_pl"]
      ]
    }
  ],
}
```

```
"out_doctype_list": [
{
"doc_type_id": "<output_data_doctype_version>",
"doc_type_version": "1.0"
}
], ,
}
```

- iii. **jarvis_byods_config.json**: Jarvis connectivity configuration needs to be provided here. Update this file based on the values that you have provided in the previous two files (**jarvis_byods_config_batch.json** and **jarvis_byods_config_speed.json**).



The outline code for batch application is provided with full documentation in `batch_main.py`.

The outline code for speed application is provided with full documentation in `speed_main.py`.

Use `\jarvis_byods\jaf.zip` (jarvis SDK) as a dependency in the project while developing your batch or speed application.

7. Set the app folder path (example, custom app) as the Python working directory for the Python file.
8. Edit **batch_main.py** or **speed_main.py** with the code that you have written for your batch or speed application.
9. Run the edited python file (`batch_main.py` or `speed_main.py`) to test the logic of your code using spark local.
 - a. For debugging speed jobs, refer the following steps:
 - i. [Download](#) the spark kafka connector jar assembly (spark-streaming-kafka-0-8-assembly_2.11 or 2.2.0) .
 - ii. Copy this jar assembly to the `%SPARK_HOME%/jars` folder.
 - iii. Set the following command in the script parameters for the spark streaming starting module:


```
--jars spark-streaming-kafka-0-8-assembly_2.11-2.2.0.jar  
pyspark-shell
```
10. Use the following command to create the zip file (for example, custom_app.zip) to be onboarded to Jarvis production system once the application is developed and tested on spark local:


```
python package.py
```
11. Onboard production phase product, tenant and required doctypes:
 - a. [Use Jarvis API to On-board Product](#)
 - b. [Use Jarvis API to On-board Tenant](#)
 - c. [Use Jarvis API to On-board input and output docTypes](#)

12. On-board Data Science applications for docType. Use the zip file that you have created in step 10.
 - a. [For Speed jobs](#)
 - b. [For Batch jobs](#)
13. [Use the CA Jarvis Dashboard to visualize results.](#)

How to Use JAF SDK?

JAF SDK contains APIs for registering and configuring your data science models. The following procedures outline the flow of invocations of the APIs while creating the Batch layer and speed layer applications.

Flow of Invocation of APIs for Creating Batch Job

1. Initialize JAF SDK.



JAF SDK will invoke the `cache_loader` method and the return value is considered as custom cache. Custom cache will be wrapped with the other job-specific information and then broadcast by the SDK.

```
global app_cache_broadcast
app_cache_broadcast = batch_jaf.init(cache_loader)
```

2. Load Data from batch storage layer.

```
data_df = batch_jaf.get_spark_df_for_batch(doctype, doctype_version)
```

3. Analyze and process the data.

4. Store results or Spark Model or pipeline by following one of the following options:

- a. Store results: API to ingest results/data to Jarvis

```
batch_jaf.store_results(result_df, out_doctype, out_doctype_version)
```

- b. Do the following steps:

- i. Create a Spark pipeline/model and store for usage in speed layer:

```
batch_jaf.store_spark_pipeline(result_df, "unique_id", False)
```

- ii. Notify speed layer about new data:

```
batch_jaf.publish_model_update_notification()
```

- c. Do the following steps:

- i. Create a custom object and store for usage in speed layer:

```
batch_path = batch_jaf.get_models_base_hdfs_folder() + "custom1"
result_df.write.parquet(batch_path, "overwrite")
```

ii. Notify speed layer about new data:

```
batch_jaf.publish_model_update_notification()
```

Flow of Invocation of APIs for Creating Batch Job

1. Intialize JAF SDK and create custom cache, if required:

```
global app_cache_broadcast
app_cache_broadcast = speed_jaf.init(cache_loader)
```

2. Load Model/pipeline if required(if produced by batch layer) :

```
global spark_pipeline
spark_pipeline = speed_jaf.get_spark_pipeline("unique_id")
```

3. Start streaming listner by passing method which will process incoming data:

```
speed_jaf.start(process_handler)
```

JAF API Reference

Here is a list of Batch and Speed methods with brief descriptions:

- [Batch API Reference](#)
 - [jaf.batch_jaf.init \(build_cache\)](#)
 - [jaf.batch_jaf.get_config_contents \(file_name\)](#)
 - [jaf.batch_jaf.get_spark_df_for_batch \(doctype, doctype_version\)](#)
 - [jaf.batch_jaf.store_results \(results_df, out_doctype, out_doctype_version, priority=None\)](#)
 - [jaf.batch_jaf.get_models_base_hdfs_folder \(\)](#)
 - [jaf.batch_jaf.store_spark_pipeline \(spark_pipeline, unique_id \)](#)
 - [jaf.batch_jaf.notify_new_models\(\)](#)
- [Speed API Reference](#)
 - [jaf.speed_jaf.init \(build_cache\)](#)
 - [jaf.speed_jaf.start \(process_handler\)](#)
 - [jaf.batch_jaf.store_results \(results_df, out_doctype, out_doctype_version, priority=None\)](#)
 - [jaf.speed_jaf.get_config_contents \(file_name\)](#)
 - [def jaf.speed_jaf.get_models_base_hdfs_folder \(\)](#)
 - [def jaf.speed_jaf.get_spark_pipeline \(unique_id\)](#)

Batch API Reference

Method	Description
Description:	
jaf. init (build_cache)	

Method	Description
	<p>All Jarvis Batch Jobs must call this method to initialize the JAF SDK.</p> <p>This method initializes Spark and builds cache required for processing.</p> <p>Algorithm-specific cache built as a spark broadcast variable will be returned.</p> <p>Parameters:</p> <p>build_cache: Method which can build a job specific cache object which needs to be broadcast(spark broadcast) by SDK.</p> <p>The object returned by this method will be wrapped with other Job details and returned as a spark broadcast.</p> <p>Returned object needs to be serializable and will be available as a spark broadcast variable.</p> <p>Return Value:</p> <p>Broadcast object, which can be retained at Algorithm Layer and used when algorithm specific configuration needs to be used</p> <p>Return Type:</p> <p>jaf.cache.AlgorithmCache</p>
<code>.batch_jaf.get_config_contents .e_name)</code>	<p>Description:</p> <p>Fetches contents of specific config file onboarded. This can be used by the Algorithm layer to get content of files onboarded.</p> <p>Parameters:</p> <p>file_name: Name of the configuration file.</p> <p>Return Value:</p> <p>Content of the configuration file.</p> <p>Return Type:Text</p>
<code>jaf.get_spark_df_for_batch doctype_version)</code>	<p>Description:</p> <p>Fetches data from Batch storage as Spark Dataframe for a given product, doctype, and doctype version. Product id is as defined for this job during onboarding.</p> <p>This method converts Avro present in storage to Spark Dataframe.</p> <p>Spark Dataframe will have columns with fields present in body of ingested json.</p>

Method	Description									
	<p>Tenant id is present in dataframe as column @tenant_id</p> <p>Datatype of each filed is retained in Dataframe columns</p> <p>Example: Ingested Json file:</p> <pre><i>{ "documents": [{ "header": { "doc_type_id": "tweet", "doc_type_version": "1.0", "product_id": "twitter", "tenant_id": "tenant" }, "body": [{ "friends_count": "161", "followers_count": "35" }] }] }</i></pre>									
	<table><tr><th>enant_id</th><th>followers_count</th><th>friends_count</th></tr><tr><td>ant</td><td>35</td><td>161</td></tr></table>	enant_id	followers_count	friends_count	ant	35	161			
enant_id	followers_count	friends_count								
ant	35	161								
	<p>Parameters:</p> <ul style="list-style-type: none">▪ doctype: Jarvis doctype of data to be fetched.▪ doctype_version: jarvis doctype version of the data to be fetched.									
	<p>Return Value:</p> <p>Spark dataframe with columns and its data type.</p>									
	<p>Return Type:</p> <p>As per the json body definition.</p>									
jaf. store_results (results_df, type, out_doctype_version, tenant_id)	<p>Description:Stores result dataframe in Jarvis.</p> <p>Input is a Spark Dataframe where columns are fields present in body on ingested json</p> <p>tenant id should be present in dataframe as column @tenant_id</p> <p>Example: If spark dataframe is as follows:</p> <table><tr><th>enant_id</th><th>followers_count</th><th>friends_count</th></tr><tr><td>ant</td><td>35</td><td>161</td></tr><tr><td>ant</td><td>34</td><td>162</td></tr></table> <p>Each row above will be converted to Jarvis Json with following format.</p> <p>Data type of each format is retained in Json.</p> <p>Only Jarvis allowed datatypes can be in columns</p>	enant_id	followers_count	friends_count	ant	35	161	ant	34	162
enant_id	followers_count	friends_count								
ant	35	161								
ant	34	162								

Method	Description
	<pre><i> { "documents": [{ "header": { "doc_type_id": "tweet", "doc_type_version": "1.0", "product_id": "twitter", "tenant_id": "tenant" }, "body": [{ "friends_count": "161", "followers_count": "35" }] }] } </i></pre> <p>Parameters:</p> <ul style="list-style-type: none"> ▪ results_df: Input result spark dataframe ▪ out_doctype: Jarvis doctype where results need to be stored ▪ out_doctype_version: Jarvis doctype version where results need to be stored ▪ priority: optional field. Priority of jarvis document to be ingested <p>Return Value:</p> <p>Return Type:</p>
<code>.batch_jaf. t_models_base_hdfs_folder ()</code>	<p>Description:Storage Folder (in HDFS) where the jobs can store intermediate data and models.</p> <p>Entire folder is exposed per product and further internal directories can be created by Jobs as per discretion.</p> <p>The same folder is available for the Speed jobs that are dependent on the batch job.</p> <p>Parameters: None</p> <p>Return Value:</p> <p>HDFS folder path</p> <p>Return Type:</p>
<code>jaf. store_spark_pipeline ipeline, unique_id)</code>	<p>Description:Stores a Spark Machine Learning (ML Pipeline) with a unique id. This pipeline built can later be used speed Job by loading with same unique_id.</p> <p>Parameters:</p> <ul style="list-style-type: none"> ▪ spark_pipeline: Spark pipeline to be persisted

Method	Description
	<ul style="list-style-type: none"> ▪ <code>unique_id</code>: unique id for the pipeline <p>Return Value: None</p> <p>Return Type:</p>
<code>.batch_jaf.notify_new_models ()</code>	<p>Description: Once new Models or Spark ML Pipelines are produced by a batch job, this method notifies the the dependent modules about the changes.</p> <p>All speed Jobs that are dependent on this output will be refreshed so that they can load the new model.</p> <p>Parameters: None</p> <p>Return Value:</p> <p>Return Type:</p>

Speed API Reference

Function	Description
<code>.speed_jaf.init (build_cache)</code>	<p>Description: All Jarvis Speed Jobs need to call this function to initialize the JAF SDK.</p> <p>This will initialize spark and builds cache required for processing. Algorithm specific cache built as a spark broadcast variable will be returned.</p> <p>Parameters:</p> <p><code>build_cache</code>: Method which can build a job specific cache object which needs to be broadcast(spark broadcast) by SDK. The object returned by this function will be wrapped with other Job details and returned as a spark broadcast Returned object needs to be serializable and will be available as a spark broadcast variable.</p> <p>Return Value:</p> <p>Broadcast object which can be retained at Algorithm Layer and used when algorithm specific configuration needs to be used</p> <p>Return Type:</p> <p><code>jaf.cache.AlgorithmCache</code></p>
<code>.jaf.start (process_handler)</code>	<p>Description:</p>

Function	Description						
	<p>Algorithm layer will invoke this function to start spark streaming.</p> <p>Algorithm layer will pass the function which implements what processing needs to be done on data. The computation paradigm is spark streaming where incoming data is microbatched and given to function provided by Algorithm Layer for processing as a data frame.</p> <p>Spark Dataframe will have columns with fields present in body of ingested json. Tenant id is present in dataframe as column @tenant_id Datatype of each field is retained in Dataframe columns Example: If ingested Json is:</p> <pre><i>{ { "documents": [{ "header": { "doc_type_id": "tweet", "doc_type_version": "1.0", "product_id": "twitter", "tenant_id": "tenant" }, "body": [{ "friends_count": "161", "followers_count": "35" }] }] } }</i></pre> <table><tr><th>tenant_id</th><th>followers_count</th><th>friends_count</th></tr><tr><td>tenant</td><td>35</td><td>161</td></tr></table> <p>Parameters:</p> <p>process_handler: Function which can handle spark dataframe. This needs to store the results using the store API internally.</p> <ul style="list-style-type: none">def process_handler(data_df):jaf.store_results(result_df, out_doctype, out_doctype_version) <p>Return Value: None</p> <p>Return Type:</p>	tenant_id	followers_count	friends_count	tenant	35	161
tenant_id	followers_count	friends_count					
tenant	35	161					
	Description:						

Function	Description									
jaf. store_results (results_df, type, out_doctype_version, lone)	<p>Stores result dataframe in Jarvis.</p> <p>Input is a Spark Dataframe where columns are fields present in body on ingested json</p> <p>tenant id should be present in dataframe as column @tenant_id</p> <p>Example: If spark dataframe is as follows:</p> <table><thead><tr><th>enant_id</th><th>followers_count</th><th>friends_count</th></tr></thead><tbody><tr><td>ant</td><td>35</td><td>161</td></tr><tr><td>ant</td><td>34</td><td>162</td></tr></tbody></table> <p>Each row above will be converted to Jarvis Json with following format.</p> <p>Data type of each format is retained in Json.</p> <p>Only Jarvis allowed datatypes can be in columns.</p> <pre><i>{ "documents": [{ "header": { "doc_type_id": "tweet", "doc_type_version": "1.0", "product_id": "twitter", "tenant_id": "tenant" }, "body": [{ "friends_count": "161", "followers_count": "35" }] }] }</pre> <p>Parameters:</p> <ul style="list-style-type: none">▪ results_df: Input result spark dataframe▪ out_doctype: Jarvis doctype where results need to be stored▪ out_doctype_version: Jarvis doctype version where results need to be stored▪ priority: optional field. Priority of jarvis document to be ingested <p>Return Value:</p> <p>Return Type:</p>	enant_id	followers_count	friends_count	ant	35	161	ant	34	162
enant_id	followers_count	friends_count								
ant	35	161								
ant	34	162								
jaf. get_config_contents (e)	<p>Description:Fetches contents of specific config file onboarded.</p> <p>This can be used by Algorithm layer to get content of files files onboarded.</p> <p>Parameters:</p>									

Function	Description
	file_name: Name of the configuration file.
	Return Value:
	Content of the configuration file.
	Return Type:
	Text
f jaf.speed_jaf. t_models_base_hdfs_folder ()	Description:
	Storage Folder(in HDFS) where the Jobs can store intermediate data and models. Entire folder is exposed per product and further internal directories can be created by Jobs.
	Parameters: None
	Return Value:
	HDFS folder path.
	Return Type:
f jaf.speed_jaf. get_spark_pipeline unique_id)	Description: Retrieve stored Spark Based model from batch storage.
	Identifier used to store can be used while retrieving the model again
	Parameters:
	unique_id: unique id for the Spark ML pipeline used during batch storage
	Return Value:
	Spark ML pipeline object
	Return Type:

Troubleshooting

- Installation
 - Error while installing Jarvis.
 - What are the configurations I should make after installing Oracle Java 1.8.0_112?
 - I am getting XMLScriptReader error while installing CA Jarvis.
 - How to start and stop Jarvis services?
 - Why I am getting bad certificate error in ESUtils log when I install Jarvis by enabling two-way SSL (ESUtils on one machine and other Jarvis components are on another machine)?
 - I am using an Elasticsearch cluster setup. What should I do if jarvis_metadata and jarvis_kron indices are not created and no document (Analytics Schema) is present in jarvis_metadata?
 - I am getting the bad interpreter error while running prepareMachineAsRoot.sh and jarvisInstaller.sh.
 - What are the different error codes that appear during installation on plain virtual machine and how to resolve the errors?
 - How to check which version of CA Jarvis that I am using?
- Log Files
 - Where can I find log files for CA Jarvis?
- Onboarding and Ingestion
 - Why I am getting the following "400 Not Found" error message with the following content when trying to onboard a product?
 - Why I am getting 5XX errors while onboarding a product?
 - Why I am getting 4XX errors while onboarding a product?
 - Why does the Ingestion API returns a 503 response?
 - How do I validate whether Ingestion is working fine or not?
 - Why I am getting 4XX errors during ingestion?
 - Why my data ingestion failed?
 - I want to do onboarding and ingestion in my SSL-enabled setup. What all should I take care for SSL?
 - I am using Postman client for onboarding and ingestion in an SSL-enabled setup. What all should I take care for SSL?
- Verifier
 - How do I check the number of documents processed by Verifier?
 - How do I validate whether Verifier is working fine or not?
 - Why the data verification failed?
- Indexer
 - How do I check the number of documents processed by indexer?
 - Why the indexer failed?
- ElasticSearch
 - How to check if Elastic cluster nodes are properly configured or not?
 - How to check the health of a cluster?
 - What do the colors (green, yellow, and red) indicate?:
 - Why I am facing a drop in the ingestion rate in my production environment?
- Kafka
 - How do I check whether Kafka is clustered or not?

- Kafka messages are not being processed by Veifier, Indexer, or Ingestion services?
- Why I am getting 5XX errors?
- Why I am getting 4XX errors?
- Data Purge
 - Where can I find the Data Purge logs?
 - How do I get to know if Data Purge has been initiated on any product/doc_type?
 - How do I validate what retention period is used if any tenant does not have retention period defined?
 - How do I ensure that Data Purge has been successfully completed?
 - Does Data Purge delete the index to which data is currently getting ingested?
- CA Jarvis Dashboard
 - What are the different error codes that I get while using CA Jarvis Dashboard APIs?

Installation

Problem

Error while installing Jarvis.

Verifying JVM...No Java virtual machine could be found from your PATH environment variable. You must install a VM prior to running this program.

Solution

Install Oracle Java 1.8.0_112 (JDK or JRE)

Problem

What are the configurations I should make after installing Oracle Java 1.8.0_112?

Solution

Complete the following steps to export JAVA_HOME environment variable to the Java version (1.8.0_112). This procedure assumes that Oracle Java 1.8.0_112 is already installed on /home/jarvis/jdk1.8.0_112.

1. Run the following command:

```
vi ~/.bashrc
```

2. Add the following line to the bashrc file, and press the **Esc** key in the keyboard, type **:wq**, and press **Enter** to save the changes made to the bashrc file.

```
export JAVA_HOME=/home/jarvis/jdk1.8.0_112
```

3. Run the following command to reflect the changes:

```
source ~/.bashrc
```

4. Verify that the installer uses Java and not open JDK by executing the following command:

```
java -version
```

You must get an output like the following:

```
java version "1.8.0_112"
Java(TM) SE Runtime Environment (build 1.8.0_112-b15)
Java HotSpot(TM) 64-Bit Server VM (build 25.112-b15, mixed mode)
```

Problem

I am getting XMLScriptReader error while installing CA Jarvis.

```
XMLScriptReader: unable to parse the provided script file. File may have been
invalid...
java.lang.NullPointerException
    at Flexeraan8.am(Unknown Source)
    at Flexeraan8.ak(Unknown Source)
    at Flexeraan8$l.aa(Unknown Source)
    at Flexeraair.ad(Unknown Source)
    at Flexeraan8.ac(Unknown Source)
    at com.zerog.ia.installer.LifecycleManager.b1(Unknown Source)
```

Solution

Execute the following commands one-by-one, and try starting the installation again.

To unset the DISPLAY variable, if set:

```
unset DISPLAY
```

If you have xauth installed on your node, run the following command to remove any xauth-related authorizations:

```
xauth remove
```

Problem

How to start and stop Jarvis services?

Solution

For Non-Root Sudo Users:

Command Format for starting and stopping services:

```
sudo systemctl <command><serviceName>
```

Command Format for checking the status of services:

```
systemctl <command> <serviceName>
```

Where <command> can be one of the following:

```
start
stop
status
```

And, <serviceName> can be one of the following:

ca_zookeeper.service should be started before starting **ca_kafka.service**. After starting **ca_kafka.service**, the rest of the services can be started in any order.

```
ca_zookeeper.service
ca_kafka.service
ca_elasticsearch.service
ca_schema-registry.service
ca_tomcat.service
ca_verifier-ingest.service
ca_indexer-ingest.service
ca_esutils.service
```

For Non-Root-Non-Sudo Users:**Command for starting the Jarvis services:**

ca_zookeeper.service should be started before starting **ca_kafka.service**. After starting **ca_kafka.service**, the rest of the services can be started in any order.

```
./startElasticsearch.sh
./startZookeeper.sh
./startkafka.sh
./startSchemaRegistry.sh
./startIndexerIngest.sh
./startIndexerUpdate.sh
./startVerifierIngest.sh
./startVerifierUpdate.sh
./startESUtils.sh
./startTomcat.sh
```

Command for stopping the Jarvis services:

```
./stopElasticsearch.sh
./stopZookeeper.sh
./stopkafka.sh
./stopSchemaRegistry.sh
./stopIndexerIngest.sh
./stopIndexerUpdate.sh
./stopVerifierIngest.sh
./stopVerifierUpdate.sh
./stopESUtils.sh
./stopTomcat.sh
```



Use the following command to verify that the Jarvis services are up and running:

```
ps -eaf |grep <name of the service>
```



Verify that `vm.max_map_count=262144` in `/etc/sysctl.conf`. If the value is different or blank, ElasticSearch will not restart after the reboot. You must manually update the value.

Run the following commands to set the `vm.max_map_count`. These commands require sudo access:

```
sysctl -q -w vm.max_map_count=262144
echo "vm.max_map_count=262144" >> /etc/sysctl.conf
```

Problem

Why I am getting bad certificate error in ESUtils log when I install Jarvis by enabling two-way SSL (ESUtils on one machine and other Jarvis components are on another machine)?

Solution

1. Ensure that Truststore.jks, Ca-cert, and Ca-key should be same in both the machines.
2. Customize and run the following command on both the machines to import ca-cert to Java security path:

```
keytool -import -alias name> -file ca-cert -keystore </Java security path>
/cacerts -storepass <password>
```

Example:

```
keytool -import -alias n2 -file ca-cert -keystore /jarvis/jdk1.8.0_112/jre
/lib/security/cacerts -storepass changeit
```

3. Restart ESUtils.

Problem

I am using an Elasticsearch cluster setup. What should I do if jarvis_metadata and jarvis_kron indices are not created and no document (Analytics Schema) is present in jarvis_metadata?

Solution:

If metadata is not created, run the following commands from the USER_INSTALL_DIR/scripts directory to create the missing indices:

```
./metadata_template.sh
```

Problem

I am getting the bad interpreter error while running prepareMachineAsRoot.sh and jarvisInstaller.sh.

```
../bash^M: bad interpreter:
```

Solution

Run the following commands one-by-one:

```
yum -y install dos2unix (This command requires root access.)
```

```
dos2unix ./<script-name>.sh
```

Problem

What are the different error codes that appear during installation on plain virtual machine and how to resolve the errors?

Solution

The following table describes the errors that might appear during the installation of CA Jarvis on plain virtual machines:



Installation-related logs: `<USER_INSTALL_DIR>/_CA Analytics_installation/Logs`

CA Jarvis Service logs

CA Jarvis Service logs (onboarding, ingestion, zookeeper, kafka, elasticsearch, verifier, indexer, esutils, and so on) : `<USER_INSTALL_DIR>/logs`

Error Code	Error Message	Resolution
100	Aborting Installation: Jarvis cannot be installed as a root user.	Create and use a non-root user for installing CA Jarvis.
101	Aborting Installation: Kafka Precheck Failed.	Check if valid IP or hostname is configured for Kafka installation details in the analyticsInstaller.properties file.
102	Installation of CA Jarvis has been completed with errors.	Check installer logs for more details.
103	Aborting Installation: Kafka and Zookeeper are not started successfully.	<ol style="list-style-type: none"> 1. Ensure that rest of the Kafka and Zookeeper services in the cluster are up and running. 2. Restart the Kafka Zookeeper service in the machine where the installation has failed.
104	Installation of CA Jarvis has been completed with errors—SchemaRegistry is not started successfully.	Check SchemaRegistry logs for more details.
111	Aborting Installation: Elasticsearch Precheck Failed.	Check if valid IP or hostname is configured for Elasticsearch installation details in the analyticsInstaller.properties file.
112	Aborting Installation: Elasticsearch installation failed.	Check installer logs for more details.
113	Installation of CA Jarvis has been completed with errors.	Check Elasticsearch logs for more details.
121	Aborting Installation: Tomcat Precheck Failed.	Check if valid IP or hostname is configured for Tomcat installation details in the analyticsInstaller.properties file.

Error Code	Error Message	Resolution
122	Aborting Installation: Tomcat installation failed.	Check installer logs for more details.
123	Aborting Installation: APIs installation failed.	Check installer logs for more details.
124	Aborting Installation: KRON installation failed.	Check installer logs for more details.
125	Installation of CA Jarvis has been completed with errors.	Check catalina.out for more details.
131	Aborting Installation: Indexer Precheck Failed.	Check if valid IP or hostname is configured for Indexer installation details in the analyticsInstaller.properties file.
132	Aborting Installation: Indexer installation failed.	Check installer logs for more details.
133	Installation of CA Jarvis has been completed with errors.	Check Indexer.logs for more details.
141	Aborting Installation: Verifier Precheck Failed.	Check if valid IP or hostname is configured for Verifier installation details in the analyticsInstaller.properties file.
142	Aborting Installation: Verifier installation failed.	Check installer logs for more details.
143	Installation of CA Jarvis has been completed with errors.	Check Verifier.logs for more details.
151	Aborting Installation: ESUtils Precheck Failed.	Check if valid IP or hostname is configured for ESUtils installation details in the analyticsInstaller.properties file.
152	Aborting Installation: ESUtils installation failed.	Check installer logs for more details.
153	Installation of CA Jarvis has been completed with errors.	Check utils.logs for more details.
201	Aborting Installation: Make sure to pass analyticsInstaller.properties as an argument and try again.	Pass analyticsInstaller.properties as an argument while invoking installation.
202	Aborting Installation: Invalid keystore path.	Check if valid keystore certificate path is configured in the analyticsInstaller.properties file.
203	Aborting Installation: Invalid truststore path.	Check if valid truststore certificate path is configured in the analyticsInstaller.properties file.

Problem

How to check which version of CA Jarvis that I am using?

Solution

Run the cat command over the file named VERSION located under USER_INSTALL_DIR/apis/:

1. Change to USER_INSTALL_DIR/apis.
2. Run the following command:
`cat VERSION`

Log Files

Problem

Where can I find log files for CA Jarvis?

Solution

- Installation-related logs: `<USER_INSTALL_DIR>/_CA Analytics_installation/Logs`
- CA Jarvis Service logs (onboarding, ingestion, zookeeper, kafka, elasticsearch, verifier, indexer, esutils, and so on) `<USER_INSTALL_DIR>/logs`

Onboarding and Ingestion

Use the following conventions while querying Elasticsearch in SSL or non-SSL setup:



For non-SSL:

```
curl -XGET 'http://localhost:9200/'
```

For SSL:

```
curl -XGET 'https://localhost:9200/' --cacert <certificate of CA>
```

Problem

Why I am getting the following "400 Not Found" error message with the following content when trying to onboard a product?

```
{"_message": "Invalid URL.", "_code": "1019"}
```

Solution

This is because of the invalid content type in the request header. Following is the correct header value for Content-type:

```
application/json
```

Problem

Why I am getting 5 XX errors while onboarding a product?

Solution

Check the following one-by-one:

1. Check for connectivity issues with Elasticsearch:
 - a. Check the apis.properties file in the config folder.
 - b. Check if the ES hostname and port is correct by running the following command:

```
curl -XGET 'http://<ES_hostname>:<port>/_cluster/health?pretty=true'
```

You need to get a reply and status value must be green.

2. Check configured Elasticsearch URL and Elasticsearch Cluster Name. The cluster name should match the name in apis.properties.

3. Check if metadata.sh was ran and jarvis_metadata was created. See if there is an index jarvis_metadata by running the following command:

```
curl -XGET 'http://<ES_hostname>:<port>/jarvis_metadata?pretty=true'
```

The reply should not contain IndexNotFoundException.

Problem

Why I am getting 4 XX errors while onboarding a product?

Solution

Check the following one-by-one:

1. Check for bad URL and correct the errors.
2. Check for bad Data - product/tenant/mapping. Ingestion

Problem

Why does the Ingestion API returns a 503 response?

Solution

The response code 503 usually indicates that Kafka is down or not reachable. Confirm by checking the ingestion log files. Also, Tomcat may be running with full load. Increase the maxConnection/maxThreads configuration in Tomcat to handle more connections.

Problem

How do I validate whether Ingestion is working fine or not?

Solution

Run CA Jarvis Healthcheck.

Problem

Why I am getting 4 XX errors during ingestion?

Solution

Check the following one-by-one:

1. Check for bad URL
2. Check for bad Data - ingestion format.

Problem

Why my data ingestion failed?

Solution

1. Ingested data body could be malformed and discarded. Check the Verifier logs.
2. Ingestion could not post to "unverified" kafka topic. Check API services(Ingestion logs).

Problem

I want to do onboarding and ingestion in my SSL-enabled setup. What all should I take care for SSL?

Solution

Ensure that you have followed the following points:

- Use **8443** as the <port number>.
- Use **https** as the scheme in the URL. http as the scheme in the URL.
- Customize and use the following command for querying Elasticsearch:

```
curl -XGET 'https://<ES_hostname>:<port>/' --cacert <certificate of CA>
```

Problem

I am using Postman client for onboarding and ingestion in an SSL-enabled setup. What all should I take care for SSL?

Solution

Complete the following procedure:

1. Customize the following URL and open it in either Safari or in Google Chrome browser:

`https://<FQDN>:8443`

The browser pops up a dialog box.

2. Click **Show Certificate**, expand **Trust**, and select **Always Trust** from the dropdown list.
3. Click Ok.
Certificate is added to your Certificates KeyChain Store. You are ready to on-board and ingest document from Postman client.

Verifier

Problem

How do I check the number of documents processed by Verifier?

Solution

The Verifier logs show how many messages it has consumed per minute from the *unverified* topic. That can be used to measure the performance of Verifier.

Problem

How do I validate whether Verifier is working fine or not?

Solution

The ingested document is written to verified or patchverified topic on Kafka. Reason for failure will be written to Verifier logs or to nohup.out(depending upon config). Execute the following command to verify that the Verifier is up and running:

```
ps -eaf |grep verifier
```

This command shows if verifier-ingest or verifier-update processes are up and running.

If Verifier has stopped or if you want to start it, change to `<USER_INSTALL_DIR>/bin` and execute the following command:

```
./startVerifierIngest.sh
```



verifier-update will be running only if use_avro_schema=false.

Problem

Why the data verification failed?

Solution

Following can be the possible reasons:

- Data could not be read from "unverified" kafka topic.
- Data could not be posted to "verified" kafka topic.
- Ingested data does not comply to the mapping scheme.

Indexer

Problem

How do I check the number of documents processed by indexer?

Solution

The indexer logs show how many messages it has consumed from the *verified* topic and pushed to Elasticsearch per minute. That can be used to measure the performance of indexer.

Problem

Why the indexer failed?

Solution

Following can be the possible reasons:

- Indexer could not write to elasticsearch indices.
- Indexer could not read from "verified" kafka topic.

ElasticSearch

Use the following conventions while querying Elasticsearch in SSL or non-SSL setup:



For non-SSL:

```
curl -XGET 'http://localhost:9200/<query>'
```

For SSL:

```
curl -XGET 'https://localhost:9200/<query>' --cacert <certificate of CA>
```

Problem

How to check if Elastic cluster nodes are properly configured or not?

Solution

Log in to the remote host where the Elasticsearch is running, and run the following command:

```
curl -s -XGET '/_cat/nodes'
```

Problem

How to check the health of a cluster?

Solution

Run the following command to use the cluster health API to get the health status of a cluster:

```
curl -XGET http://localhost:9200/_cluster/health?pretty=true
```

What do the colors (green, yellow, and red) indicate?:

Green

Indicates that the cluster is 100% functional.

Yellow

Indicates that all primary shards are active, but not all replica shards are active.

Red

There can be many reasons for the Red state. Following are some of the common reasons:

1. At least one primary shard (and all of its replicas) is missing or unassigned. This means that you are missing data. So, searches will return partial results, and indexing into that shard will return an exception.
2. Nodes are down.
3. Disk is full. Try to purge some outdated data or try to scale up the nodes.

4. Also, check the following:
 - a. Is cluster set up correctly?
 - b. Has Onboarding Started?
 - c. Default shard size specified in `apis.properties`.

Problem

I am running Elasticsearch instance in a single node setup. I am getting the following result when I checked the status:

```
{
  "cluster_name": "mylocal",
  "status": "red",
  "number_of_nodes": 1,
  "assigned_shards": 4,
  "unassigned_shards": 126
}
```

Solution

The status is "red" and there are many unassigned shards. This is because of your default configuration of Elasticsearch server. To fix this issue, do the following steps:

1. Open your `elasticsearch.yml` file.
2. Add the following lines to the file:


```
index.number_of_shards: 1
index.number_of_replicas: 0
```
3. Restart Elasticsearch.
4. Run the following command to reflect this settings in all existing indices.


```
$ curl -XPUT 'localhost:9200/_settings' -d '{"index.number_of_replicas": 0}'
```

Problem

Why I am facing a drop in the ingestion rate in my production environment?

Solution

One of the following can cause a drop in the ingestion rate:

- A wrongly set up cluster. Ensure that the Elasticsearch cluster is set up using the Jarvis installer, and not through manual steps.
- The mappings used are inconsistent. This can cause to additional overhead on Elasticsearch to index the fields. Collaborate with the Jarvis team to ensure that the mappings are defined correctly.
- An undersized cluster, which might have filled up the capacity, can leads to a lower ingestion rate. Add more nodes to the cluster to handle more loads.

Kafka

Problem

How do I check whether Kafka is clustered or not?

Solution

Run the following commands from the Kafka home directory to check whether Kafak is clustered properly or not:

```
bin/kafka-console-consumer.sh --zookeeper --topic verified --from-beginning
```

```
bin/kafka-console-producer.sh --broker-list --topic verified
```

After running the producer command (second command), enter some message. The message should appear in the consumer console.

Problem

Kafka messages are not being processed by Veifier, Indexer, or Ingestion services?

Solution

Follow these steps:

1. Open the .properties file of the relevant service. For example, verifier.properties.
2. Search for kafka_brokers or brokers.
3. Ensure that the value (localhost:9092) set is the same as the value of listeners in the server.properties file for Kafka.

Problem

Why I am getting 5 XX errors?

Solution

Check the following one-by-one:

1. Connectivity issues with Kafka.
2. Configured KafkaURL.

Problem

Why I am getting 4 XX errors?

Solution

Check the following one-by-one:

1. URL.
2. Format of the ingestion data.

Data Purge

Problem

Where can I find the Data Purge logs?

Solution

If you are in the JarvisInstaller folder, relative path for logsis: ../jarvis/jarvisdatapurge/jarvisdatapurge/logs.

Problem

How do I get to know if Data Purge has been initiated on any product/doc_type?

Solution

Data Purge logs show on which product/doc_type it is running the purge process.

Problem

How do I validate what retention period is used if any tenant does not have retention period defined?

Solution

Data Purge uses product-level retention period. If that is not defined, Purge uses the default retention period. You can check purge logs to see the retention period that will be used.

Problem

How do I ensure that Data Purge has been successfully completed?

Solution

The logs show a message saying that "Data Purge is Completed."

Problem

Does Data Purge delete the index to which data is currently getting ingested?

Solution

No. Data might come at any time and it gets ingested to the index. So, Data Purge does not delete that index even if the data is older than the retention period.

CA Jarvis Dashboard

Problem

What are the different error codes that I get while using CA Jarvis Dashboard APIs?

Solution

<LDDS team, can you review these messages and provide the correct error messages to me? >

Error Code	Error Message	Resolution
400	Payload JSON structure not acceptable	The payload JSON file format is not acceptable. Review the JSON file format and retry. Note: The system does not change the existing dashboards.
	User Onboarding: Username does not have a valid email address.	Provide a valid email address as the value for username, and retry.
	Supplied key-value pairs are malformed.	Check the following and retry: <ul style="list-style-type: none"> Keys are supplied Key names are not misspelled
	User Onboarding: Bad request.	While creating a user, authGroup can have the following three values in the API: <ul style="list-style-type: none"> "OA" "TA"

Error Code	Error Message	Resolution
		<ul style="list-style-type: none"> ▪ "PA" <p>If other values are supplied with the request, correct it and retry.</p>
	User Onboarding:	While, listing user name, the HTTP headers are not supplied or malformed. Correct the headers and retry.
	User Onboarding: Invalid productCode, tenantCode, or username combination.	Invalid values for productCode, tenantCode, username combination. Correct the values and retry.
	User Onboarding: Malformed username. Username should be a valid email address.	Ensure that the username has a valid email address structure and retry.
	User Onboarding: Invalid password structure	The supplied password have a stipulated structure. <Is there any recommendations for the password—we should state those here.>
	User Onboarding: AuthGroup should only have the following values -- OA, PA, TA"	<p>Ensure that the authGroup parameter has one of the following values:</p> <ul style="list-style-type: none"> ▪ OA ▪ TA ▪ PA
	User Onboarding: Invalid productCode, tenantCode, userName, password, or authGroup	Invalid values for productCode, tenantCode, userName, password, or authGroup combination. Correct the values and retry.
404	Product Onboarding: Product with the productId <productCode> does not exist	Provide a valid productCode.
	Product Onboarding: Product with the productId <productCode> contains tenants and hence can't be deleted	Delete the tenants that are related to the product that you are going to delete, and retry.
	Tenant Onboarding: Tenant with the tenantCode <tenantCode> does not exist	Provide a valid tenantCode.
	Tenant Onboarding: tenantCode and/or productCode do not exist.	Provide a valid tenantCode and/or productCode.
	User Onboarding: ProductCode /tenantCode/username combination does not exist	Provide a valid tenantCode , productCode, and userName combination.
	User Onboarding: ProductCode /tenantCode combination does not exist.	Provide a valid productCode/tenantCode combination.
406	Product Onboarding: Product does not exist in Jarvis	Ensure that CA Jarvis already has a product of the same name that you are going to onboard.
	Jarvis connection failed	Ensure that the CA Jarvis instance is accessible to CA Jarvis Dashboard to successfully onboard products and tenants.
	Tenant Onboarding: Tenant does not exist in Jarvis	Ensure that CA Jarvis already has a tenant of the same name that you are going to onboard.

Error Code	Error Message	Resolution
409	Product Onboarding: Product already on-boarded	Onboard a different product.
	Product Onboarding: Product with the new productID exists	Retry updating product information with a different productCode.
	Tenant Onboarding: Tenant already on-boarded	Onboard a different tenant.
	Tenant Onboarding: Product with the new tenantId exists	Retry updating tenant information with a different tenantCode.
	User Onboarding: User already on-boarded.	The supplied username already exists within the productCode/tenantCode scope in the database. Retry with a different user name.

Product Accessibility Features

CA Technologies is committed to ensuring that all customers, regardless of ability, can successfully use its products and supporting documentation to accomplish vital business tasks.