

RiteSync

Installation, Configuration

**Table of Contents**

[RITESYNC 2](#_Toc126081743)

[Ritesync Application Server Setup 4](#_Toc126081744)

[**Java Installation** 8](#_Toc126081745)

[**Kafka Installation and configuration** 9](#_Toc126081746)

[Configuring the Kafka Server 9](#_Toc126081747)

[**Creating Systemd Unit Files and Starting the Kafka Server** 11](#_Toc126081748)

[**Ritesync Application Configuration** 14](#_Toc126081749)

[**Creating Systemd Unit File and Starting the Ritesync Server** 17](#_Toc126081750)

## RITESYNC

Strategic investments from the CTO’s office are needed to synchronize ever-increasing enterprise data and derive value from analytics while balancing it with cost-effective smart data platforms.

RiteSync bridges the data synchronization needs from Oracle Cloud Applications to enterprise data warehouse and analytics platforms like Snowflake and Power BI. It is built on a cloud-native modular architecture, enabling customers to sync Oracle Cloud Applications data natively in different formats, such as Parquet, JSON, XML and direct mapping with the database. Customers can run advanced analytics bypassing the warehousing step, allowing business users and citizen developers to build reports without any handholding.



The process of pulling data from one SaaS database and uploading it to another database is tedious and expensive due to hosts levied restrictions, which might become a hurdle or even lead to further expenditure. Oracle Cloud Applications do not allow downloading additional data than what is defined, and therefore businesses have to invest in additional solutions to address their data warehousing needs. RiteSync provides the functionality to pull data from any Oracle Cloud Applications database and upload it into the desired database. Customers have the capability to schedule data sync based on the module, object, and database of their choice with its limitations and restrictions depending on the upgrade done by Oracle.

**Features**

* Leverage your investment in data warehouse solutions using data synchronization from cloud platforms
* A PaaS solution to secure clients' confidential data in a secure, HIPAA-compliant system managed by Oracle / Customer Datacenter compute
* Cut down on capital expenditure with a flexible, pay-per-use pricing model
* Adaptor to load data directly to Oracle database / file-types
* Flexible scheduling capabilities at the module level

# Ritesync Application Server Setup

#### **Create a User with SUDO Permissions**:

* Create a user with Using the command below,

# useradd ritesync

* Give a Password for ‘ritesync’ user

# passwd ritesync

* Give entry of ritesync user in /etc/sudoers file
* Then Login to the User

# su – ritesync

#### **Nginx Installation & Configuration** :

* First Install NGINX in server Using ‘YUM’

$ sudo yum install nginx -y

$ cd /etc/nginx/

$ sudo mkdir sites-available sites-enabled

$ vi sites-available/ritesync.conf

|  |
| --- |
| server {  listen 80;  listen [::]:80;  server\_name ritesync.ritesoftware.com;  root /usr/share/nginx/html/;  index index.html;  return 301 https://ritesync.ritesoftware.com$request\_uri;  location /  {  try\_files index.html $uri $uri/ /index.html;  }  location /RiteSync{  proxy\_pass http://127.0.0.1:9098/;  }  location /index.html  {  rewrite ^/oldURL$ https://ritesync.ritesoftware.com/newURL redirect;  }  }  server {  listen 443 ssl http2 default\_server;  listen [::]:443 ssl http2;  server\_name ritesync.ritesoftware.com;  root /usr/share/nginx/html;  ssl\_certificate “<**SSL certificate path>**";  ssl\_certificate\_key "<**SSL Key path>**";  location /RiteSync{  proxy\_pass http://127.0.0.1:9098/;  }  location /  {  try\_files index.html $uri $uri/ /index.html;  }  ssl\_protocols TLSv1.2 TLSv1.1 TLSv1;  location /index.html  {  rewrite ^/oldURL$ https://ritesoftware.com.com/newURL redirect;  }  } |

* Save and exit the file
* **Create a Link File in /etc/nginx/sites-enabled**

$ ln -s /etc/nginx/sites-available/ritesync.conf /etc/nginx/sites-enabled/

* **Insert below lines in /etc/nginx/nginx.conf file**

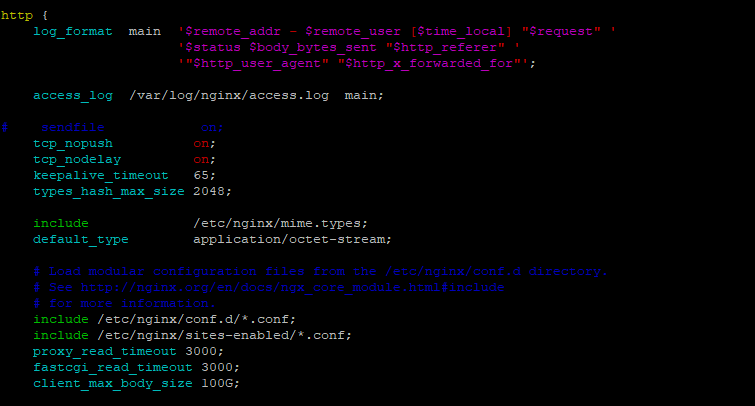
include /etc/nginx/sites-enabled/\*.conf;

proxy\_read\_timeout 3000;

fastcgi\_read\_timeout 3000;

client\_max\_body\_size 100G;

**Picture for your reference:**



* Save and Exit
* To check nginx configuration errors follow this command

$ sudo nginx -t

Text

Description automatically generated

* Now Create a directory called “**ritesync**” in the following path

$ mkdir /usr/share/nginx/html/ritesync

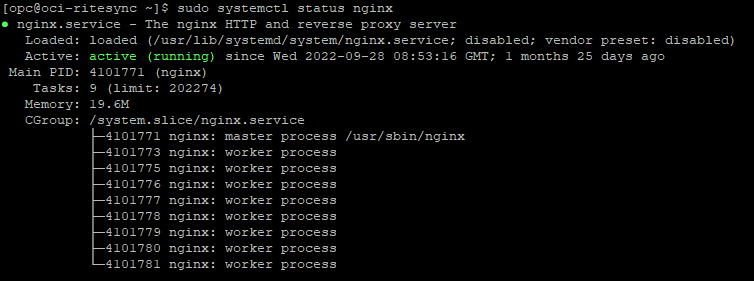
* Place the **Ritesync** UI files (CSS, JS,INDEX.HTML) in the above path using WINSCP OR SCP
* **Now start the Nginx service**

$ sudo systemctl enable nginx

$ sudo systemctl start nginx

$ sudo systemctl status nginx

**Picture for your reference:**



### **Java Installation**

$ wget <https://download.oracle.com/otn/java/jdk/14.0.2+12/205943a0976c4ed48cb16f1043c5c647/jdk-14.0.2_linux-x64_bin.rpm?AuthParam=1663670919_52c525d73484b242d2435cf552f58150>

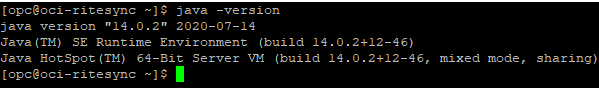
(**Get the fresh Jdk download link from oracle E- Delivery site)**

$ mv jdk-14.0.2\_linux-x64\_bin.rpm\?AuthParam\=1663670919\_52c525d73484b242d2435cf552f58150

jdk-14.0.2\_linux-x64\_bin.rpm

$ sudo yum localinstall jdk-14.0.2\_linux-x64\_bin.rpm -y

$ java -version



### **Kafka Installation and configuration**

* To download the Kafka, go to the link below.
* https://downloads.apache.org/kafka/3.2.3/kafka\_2.12-3.2.3.tgz

$ sudo mkdir /home/ritesync/kafka

$ chmod -R 777 /home/ritesync/kafka/

$ cd /home/ritesync/kafka

$ wget https://downloads.apache.org/kafka/3.2.3/kafka\_2.12-3.2.3.tgz

## **Configuring the Kafka Server**

* Now unzip the downloaded zip file.

$ tar -xvzf kafka\_2.12-3.2.3.tgz

* Create a directory for Kafka logs.

$ mkdir /home/ritesync/kafka/logs

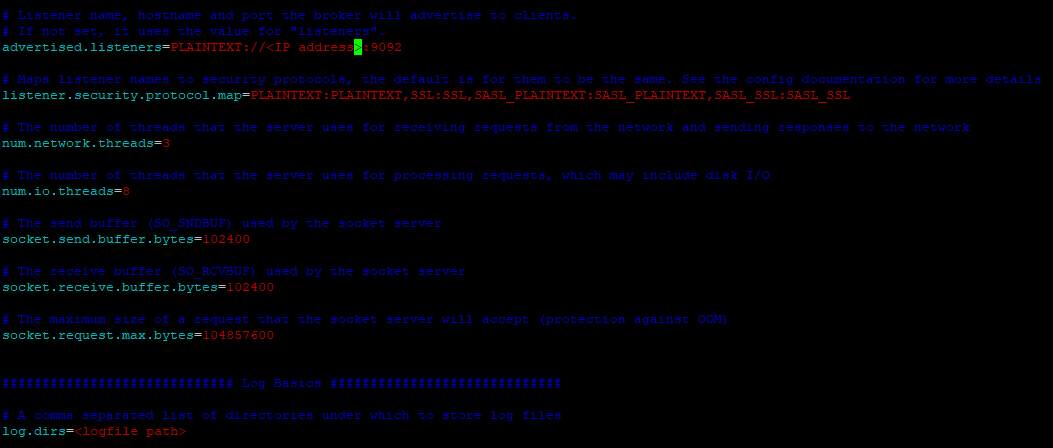
$ mkdir /home/ritesync/kafka/zookeeper

$ chmod -R 777 /home/ritesync/kafka/

* Kafka’s configuration options are specified in server.properties. Open this file with vi or your favorite editor.

$ vi /home/ritesync/kafka/config/server.properties

* Change the directory where the Kafka logs are stored by modifying the “logs.dir” property.
* And uncomment the “advertised.listener” and mention IP Address as shown below,
* Uncomment the line “listener.security.protocal.map”



* Save and close the file. Now that you’ve configured Kafka.
* Now change the “datadir” path in “zookeeper.properties file

$ vi /home/ritesync/kafka/config/zookeeper.properties

~/home/ritesync/kafka/config/zookeeper.properties

dataDir=/home/ritesync/kafka/zookeeper

### **Creating Systemd Unit Files and Starting the Kafka Server**

* In this section, you will create systemd unit files for the Kafka service. This will help you perform common service actions such as starting, stopping, and restarting Kafka in a manner consistent with other Linux services.
* Zookeeper is a service that Kafka uses to manage its cluster state and configurations. It is used in many distributed systems.

**Create the unit file for zookeeper:**

$ sudo vi /etc/systemd/system/zookeeper.service

**Enter the following unit definition into the file:**

|  |
| --- |
| **[Unit]**  **Requires=network.target remote-fs.target**  **After=network.target remote-fs.target**  **[Service]**  **Type=simple**  **User=root**  **ExecStart=/home/ritesync/kafka/bin/zookeeper-server-start.sh /home/ritesync/kafka/config/zookeeper.properties**  **ExecStop=/home/ritesync/kafka/bin/zookeeper-server-stop.sh**  **Restart=on-abnormal**  **[Install]**  **WantedBy=multi-user.target** |

**Create the unit file for Kafka:**

$ sudo vi /etc/systemd/system/kafka.service

**Enter the following unit definition into the file:**

|  |
| --- |
| **[Unit]**  **Requires=zookeeper.service**  **After=zookeeper.service**  **[Service]**  **Type=simple**  **User=root**  **ExecStart=/home/convertrite/kafka\_2.13-2.8.2/bin/kafka-server-start.sh /home/convertrite/kafka\_2.13-2.8.2/config/server.properties**  **ExecStop=/home/convertrite/kafka\_2.13-2.8.2/bin/kafka-server-stop.sh**  **Restart=on-abnormal**  **[Install]**  **WantedBy=multi-user.target** |

* Now that you have defined the units, start Kafka and zookeeper with the following commands.

$ sudo systemctl daemon-reload

$ sudo systemctl start zookeeper

$ sudo systemctl status zookeeper

Text

Description automatically generated with low confidence

$ sudo systemctl start kafka

$ sudo systemctl status kafka

Text

Description automatically generated

$ sudo systemctl enable zookeeper

$ sudo systemctl enable kafka

### **Ritesync Application Configuration**

##### **Directory Config :**

$ sudo mkdir -p ritesync/external-properties ritesync/ritesynclogs

$ sudo mkdir -p ritesynccloudintegretor/config

$ sudo mkdir -p loaderservice/config

$ sudo mkdir data

##### **Ritesync Configuration :**

* Place the Ritesync JAR in /home/ritesync/ritesync/
* After that go to external-properties dir and create a file like below,

$ vi external.properties

|  |
| --- |
| logging.file.name=/home/ritesync/ritesync/ritesynclogs/application.log  server.port=9098  # Oracle settings  context-path=/RiteSync  application-hostname=<application IP address>  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  spring.datasource.url=jdbc:mysql://<application IP address>:3306/<Database name>  spring.datasource.username=<DB Username>  spring.datasource.password= <DB password>  spring.jpa.show-sql=true  spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect |

* Save and exit the file.

##### **Cloud Integrator Configuration :**

* Place the Cloud Connector JAR in /home/ritesync/ritesynccloudintegretor
* After placing the jar Go to config dir and create a file like below,

$ cd /home/ritesync/ritesynccloudintegretor/config

$ vi application.properties

|  |
| --- |
| #security-service-url=https://ucf5-zsjk-fa-ext.oracledemos.com/xmlpserver/services/v2/SecurityService  security-service-url=/xmlpserver/services/v2/SecurityService  #catalog-service-url=https://ucf5-zsjk-fa-ext.oracledemos.com/xmlpserver/services/v2/CatalogService  catalog-service-url=/xmlpserver/services/v2/CatalogService  #report-service-url=https://ucf5-zsjk-fa-ext.oracledemos.com/xmlpserver/services/v2/ReportService  report-service-url=/xmlpserver/services/v2/ReportService  absolute-path=/~linda.cairns/ConverRiteTest  #producer-rest-url=https://<IP/Hostname>/RiteSync/kafkaproducer  datasource.hostname=<DB server IP>  datasource.username=<DB Username>  datasource.password=<DB Password>  datasource.database=<DB name>  file-upload-dir=/home/ritesync/data/  sftp-client-host=<SFTP server IP>  sftp-client-username=<SFTP Username>  sftp-client-password=<SFTP User Password>  ########BATCH-PARAMETERS  batch-size=5000  limit=5000  thread-sleep-time=60000 |

* Save and exit the file

##### **Loader service Configuration :**

* Place the Loader service JAR in /home/ritesync/loaderservice dir
* After Placing the jar go to ‘config’ dir and create a file.

$ cd /home/ritesync/loaderservice/config

$ vi application.properties

|  |
| --- |
| security-service-url=/xmlpserver/services/v2/SecurityService  catalog-service-url=/xmlpserver/services/v2/CatalogService  report-service-url=/xmlpserver/services/v2/ReportService  absolute-path=/~linda.cairns/ConverRiteTest  datasource.hostname=<DB server IP/Hostname>  datasource.username=<DB Username>  datasource.password=<DB Password>  datasource.database=<DB name>  file-upload-dir=/home/ritesync/data  sftp-client-host=<SFTP server IP>  sftp-client-username= <SFTP Username>  sftp-client-password=<SFTP server Password>  batch-size=20000  limit=20000  thread-sleep-time=60000  #MSSQL Details  mssql.url = jdbc:sqlserver://ritesync.database.windows.net:1433;database=<DB name>;user=<DB Username>;password=<DB Password> |

* Save and exit the file

### **Creating Systemd Unit File and Starting the Ritesync Server**

* In this section, you will create systemd unit file for the Ritesync service. This will helps you to perform common service actions such as starting, stopping, and restarting ritesync, cloud integrator and loader service JARS in a manner consistent with other Linux services.
* Create a directory as “service-scripts”

$ mkdir /home/ritesync/service-scripts

* Now create a bash script file in /home/ritesync/service-scripts for ritesync service.

$ vi /home/ritesync/service-scripts/ritesync.sh

|  |
| --- |
| #!/bin/bash  cd /home/ritesync/ritesynccloudintegretor  java -jar RiteSyncConnectorService-0.0.1-SNAPSHOT.jar 2>> /dev/null 1>> /dev/null &  cd /home/ritesync/loaderservice  java -jar RiteSyncLoadService-0.0.1-SNAPSHOT.jar 2>> /dev/null 1>> /dev/null &  cd /home/ritesync/ritesync  java -jar rite-sync-app-0.0.1-SNAPSHOT.jar 2>> /dev/null 1>> /dev/null |

* Save and exit the file.

$ chmod 777 /home/ritesync/service-scripts/ritesync.sh

##### **Create the unit file for RITESYNC:**

$ sudo vi /etc/systemd/system/ritesync.service

**Enter the following unit definition into the file:**

|  |
| --- |
| **[Unit]**  **Description=Rite Sync Application Service**  **[Service]**  **User=root**  **Type=simple**  **ExecStart=/home/ritesync/service-scripts/ritesync.sh**  **Restart=on-failure**  **SuccessExitStatus=143**  **TimeoutStopSec=10**  **RestartSec=5**  **[Install]**  **WantedBy=multi-user.target** |

* Save and exit the file.
* Now start the Ritesync service.

$ sudo systemctl daemon-reload

$ sudo systemctl start ritesync.service

$ sudo systemctl status

Text

Description automatically generated