HCLSoftware

HCL IntelliOps
Event Management

Collector Installation Guide

Version 1.3

Table of Contents

Table of Figures

List of Tables

Document Revision History

This guide is updated with each release of the product or when necessary.

This table provides the revision history of this Collector Installation Guide.

Date	Description
January, 2024	HCL_IEM_ v1.0_Collector_Installation_Guide
January, 2025	HCL_IEM_ v1.1_Collector_Installation_Guide
February, 2025	HCL_IEM_ v1.2_Collector_Installation_Guide
July, 2025	HCL_IEM_ v1.3_Collector_Installation_Guide

1 Preface

This section provides information about the IEM Collector Installation Guide and includes the following topics:

- Intended Audience
- About This Guide
- Related Documents
- Conventions

1.1 Intended Audience

This guide is intended for IEM administrator users for IEM collector deployments.

1.2 About this Guide

This guide provides detailed installation process of IEM Collectors.

1.3 Related Documents

The following documents can be referred to in addition to this guide for further information on the IEM platform:

- IEM Configuration Guide

1.4 Conventions

The following typographic conventions are used in this document:

Table 1 – Conventions

Convention	Element
Boldface	Indicates graphical user interface elements associated with an action, or terms defined in text or the glossary
Underlined blue face	Indicates cross-reference and links
Courier New (Font)	Indicates commands within a paragraph, URLs, code in examples, and paths including on screen text and text input from users
Numbered lists	Indicates steps in a procedure to be followed in a sequence
Bulleted lists	Indicates a list of items that is not necessarily meant to be followed in a sequence

2 IEM Collector

IEM Collector refers to effectively gathering data from diverse sources, providing a wide range of single clicks, custom integrations compliant with the industry standards for connectors and APIs. The events, data and performance connectors are developed in **Apache NiFi**. These **OOB NiFi** connectors can be leveraged for data ingestion very quickly via **IMM (Integration Management Module)** Portal

2.1 Overview for NiFi

Apache NiFi is an **open-source** dataflow system based on the concepts of flow-based programming. It supports powerful and scalable directed graphs of data routing, transformation, and system mediation logic.

NiFi has a web-based user interface for design, control, feedback, and monitoring of dataflows. It is highly configurable along several dimensions of quality of service, such as loss-tolerant versus guaranteed delivery, low latency versus high throughput, and priority-based queuing.

NiFi provides fine-grained data provenance for all data received, forked, joined cloned, modified, sent, and ultimately dropped upon reaching its configured end-state.

2.1.1 NiFi Architecture

Apache NiFi has a processor, flow controller, and web server that executes on the JVM machine. Additionally, it also includes three repositories, as shown in the figure, which are FlowFile repository, Content and Provenance repository.

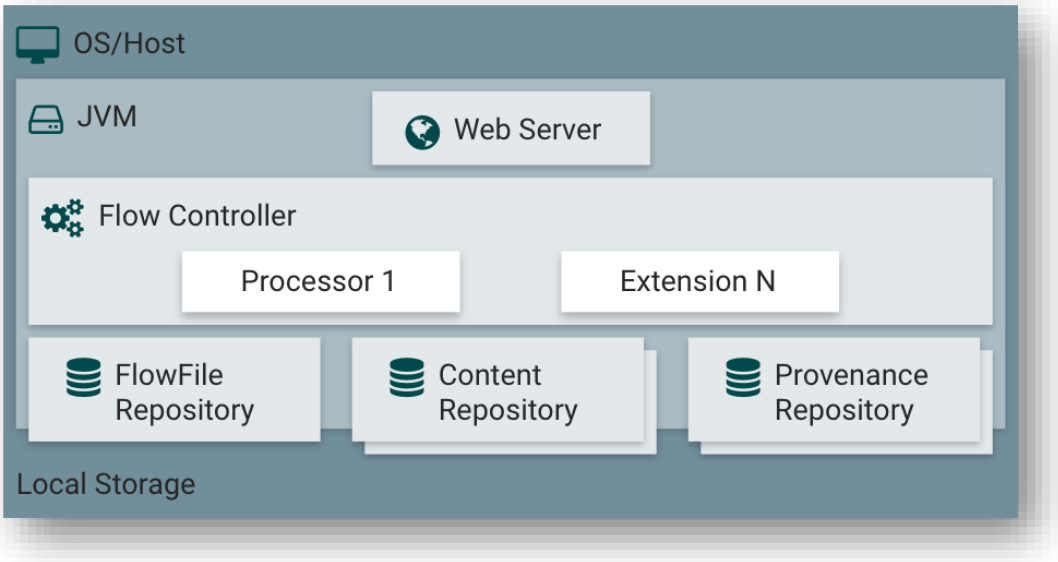


Figure 1 - Content and Provenance repository

NiFi executes within a JVM on a host operating system. The primary components of NiFi on the JVM are as follows:

- **Web Server:** The purpose of the web server is to host NiFi's HTTP-based command and control API.
- **Flow Controller:** The flow controller is the brain of the operation. It provides threads for extensions to run on and manages the schedule of when extensions receive resources to execute.
- **Extensions:** There are various types of NiFi extensions which are described in other documents. The key point here is that extensions operate and execute within the JVM.

- **FlowFile Repository:** The FlowFile Repository is where NiFi keeps track of the state of what it knows about a given FlowFile that is presently active in the flow. The implementation of the repository is pluggable. The default approach is a persistent Write-Ahead Log located on a specified disk partition.
- **Content Repository:** The Content Repository is where the actual content bytes of a given FlowFile live. The implementation of the repository is pluggable. The default approach is a simple mechanism, which stores blocks of data in the file system. More than one file system storage location can be specified to get different physical partitions engaged to reduce contention on any single volume.
- **Provenance Repository:** The Provenance Repository is where all provenance event data is stored. The repository construction is pluggable with the default implementation being to use one or more physical disk volumes. Within each location data is indexed and searchable.

NiFi is also able to operate within a cluster.

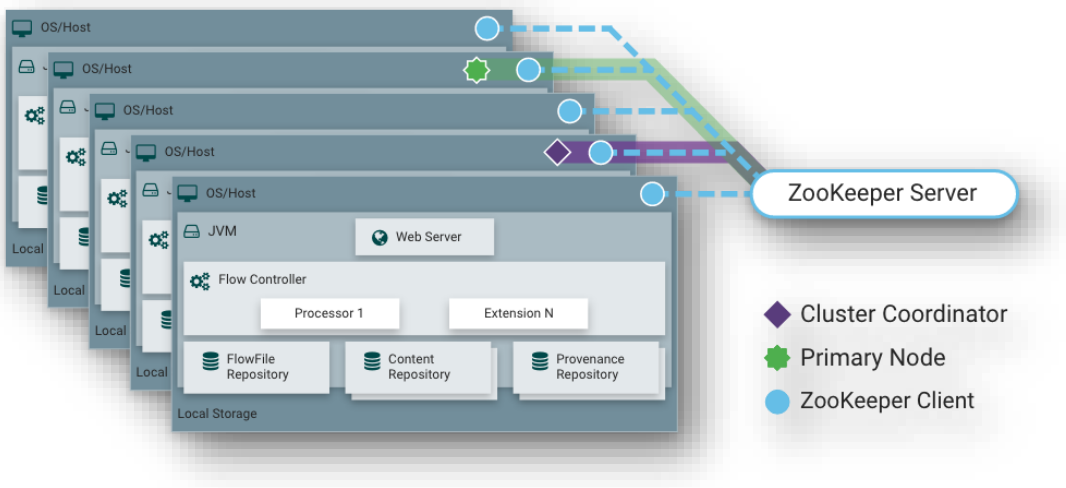


Figure 2 - Repositories

Starting with the NiFi 1.0 release, a Zero-Leader Clustering paradigm is employed. Each node in a NiFi cluster performs the same tasks on the data, but each operates on a different set of data.

Apache ZooKeeper elects a single node as the Cluster Coordinator, and failover is handled automatically by ZooKeeper. All cluster nodes report heartbeat and status information to the Cluster Coordinator. The Cluster Coordinator is responsible for disconnecting and connecting nodes.

Additionally, every cluster has one Primary Node, also elected by ZooKeeper. As a DataFlow manager, you can interact with the NiFi cluster through the user interface (UI) of any node. Any change you make is replicated to all nodes in the cluster, allowing for multiple entry points.

2.2 Prerequisites

Prerequisites are specific conditions that need to be met before initiating the configuration. Hence, mentioned below are pre-requisites for NiFi:

2.2.1 System Requirements For NiFi

- Apache NiFi can run on something as simple as a laptop, but it can also be clustered across many enterprise-class servers. Therefore, the amount of hardware and memory needed will depend on the size and nature of the dataflow involved.
- The data is stored on the disk while NiFi is processing it. So NiFi needs to have sufficient disk space allocated for its various repositories, particularly the content repository, flowfile repository, and provenance repository (see the System Properties section for more information about these repositories). NiFi has to be configured according to the following system requirements:

2.2.2 Supported NiFi Version

- NiFi 2.2

2.2.3 Supported OS for NIFI

Linux RHEL 9 (Recommended)

- Unix
- Windows
- macOS

Requires Java 21

2.2.4 Supported Web Browsers

- Microsoft Edge: Current & (Current - 1)
- Google Chrome: Current & (Current - 1)
- Safari: Current & (Current - 1)

2.2.5 Hardware Sizing Recommendation

NiFi is designed to take advantage of:

- all the cores on a machine
- all the network capacity
- all the disk speed
- many gigabytes of RAM (although usually not all) on a system

Hence, it is important that NiFi should be running on dedicated nodes. The following are the recommended server and sizing specifications for NiFi:

- Minimum of 3 nodes
- 8+ core per node (more is better)
- At least 8 GB
- 6+ disks per node (SSD or spinning)

Table 2 - Hardware Sizing Recommendation

Required Sustained Throughput	Minimum Hardware Requirement

85 events per second	• 3nodes • 4 or more cores per node • 6 or more disks per node (SSD or spinning) • 8 GB memory per node • 1 GB bonded NICs
114 events per second	• 3 nodes • 8 or more cores per node • 6 or more disks per node (SSD or spinning) • 16 GB of memory per node • 1 GB bonded NICs

2.2.6 Port Requirement for NiFi

The following ports are required for internal communication:

Port	Protocol	Direction	Source	Destination	Description	Application Name	Load Balancer
9443	TCP	Bidirectional	IMM	NiFi servers	To open NiFi UI	NiFi	LB
10443	TCP	Bidirectional	NiFi servers	NiFi servers	To run NiFi as a cluster	NiFi	
11443	TCP	Bidirectional	NiFi servers	NiFi servers	To run NiFi as a cluster	NiFi	
6342	TCP	Bidirectional	NiFi servers	NiFi servers	To run NiFi as a cluster	NiFi	
2181	TCP	Bidirectional	NiFi servers	NiFi servers	To run NiFi as a cluster	NiFi	
2888	TCP	Bidirectional	NiFi servers	NiFi servers	To run NiFi as a cluster	NiFi	
3888	TCP	Bidirectional	NiFi servers	NiFi servers	To run NiFi as a cluster	NiFi	
443/4200	TCP	Bidirectional	End User	IMM Server	Web Application	IMM	LB
4100	TCP	Bidirectional	End User	IMM Server	Web Api	IMM	LB
4300	TCP	Bidirectional	End User	IMM Server	Orchestrator Service	IMM	
5432	TCP	Bidirectional	IMM Server	IMM DB Server	Database	IMM	

2.4 NiFi Installation and Setup

Follow the steps and concise instructions given below to set up and install NiFi.

2.4.1 NiFi Collector Standalone and Cluster Installation – Auto Installation

For automatic NiFi Collector Standalone and Cluster Installation, perform the following steps:

1. **Download Installation Packages**: Download the following packages on each target NiFi server:
   - controller.zip
   - nifi_installer
2. Place both files in a working directory, e.g.:
   /HOME/NIFI_INSTALL/
3. Unzip the Controller Package i.e., controller.zip
4. Start the NiFi Installer by running the following commands:

```
chmod +x nifi_installer
./nifi_installer
```

5. Provide Configuration Details for the following prompts:

Table 3 – Prompts from Installer

Prompt	Description
Enter the cluster size (Ex: 1,3, 5...):	1 for standalone OR >1 for clustered setup
Enter remote server FQDN (Ex: hostname.domain.com):	Not Required for standalone. For cluster, it will prompt for the FQDNs based on the number of nodes.
Do you want to add load balancer to the NiFi configuration? (Yes/No):	Optional. Enter 'Yes' if you have a load balancer in front of the cluster.
Enter the load balancer IP address:	It appears only if the load balancer is selected.
Enter Service Account username:	Example: nifiadmin
Enter Service Account password:	Provide the password securely.
Enter the installation path:	Example: /opt/iemnifi

6. Do the Pre-installation Checks. The installer will:
   - Validates port availability
   - Checks system resource availability
7. You'll be asked if you want to continue. Type **yes** to continue.
8. This will complete the Installation.
9. NiFi will be installed under the specified path (e.g. /opt/iemnifi)
10. Installer will generate an admin_creds.txt file on the server containing login credentials.

2.4.1.1 Post-Installation Notes

- **NiFi UI Access**:
  - Standalone: https://<hostname>:9443/nifi
  - Cluster: https://<load-balancer>:9443/nifi (if configured)
- **Credentials**: Check admin_creds.txt file
- **Start/Stop Scripts**:
  - /opt/iemnifi/bin/nifi.sh start
  - /opt/iemnifi/bin/nifi.sh stop

2.4.2 NiFi Collector Standalone Installation - Manual

2.4.2.1 Pre-requisites for Standalone Installation

Before using Apache NiFi, the following things must be done on your system:

1. A service account must be created on all specific servers.
2. The service account must be granted sudo privileges.

Going forward, for the SNMP trap integration, use the default UDP port 162. It is necessary to make that port unprivileged.

3. Run below command to change the password of that user:

```
sysctl -w net. ipv4.ip_unprivileged_port_start=162
```

Sample Console:

```
[root@S        .P002 ~]# sysctl -w net.ipv4.ip_unprivileged_port_start=162
net.ipv4.ip_unprivileged_port_start = 162
[root@S        P002 ~]#
```

Perform all operations as NiFi admin user only.

4. Switch to “service account” user to complete the installation steps.
5. Check if Java 21 is installed on servers. If not, run the command below for JDK installation:

```
sudo yum install java-21-openjdk
```

6. Validate the version of the java. Run the below command check installed java version:

```
java -version
```

```
openjdk version "21.0.7" 2025-04-15 LTS
OpenJDK Runtime Environment (Red_Hat-21.0.7.0.6-1) (build 21.0.7+6-LTS)
OpenJDK 64-Bit Server VM (Red_Hat-21.0.7.0.6-1) (build 21.0.7+6-LTS, mixed mode, sharing)
```

7. Ensure that the hostname of each server can be resolved from all other servers, and that all servers are mutually reachable over the network.
8. Switch to “nifiadmin” user to compete the installation steps. Create directories and files required for integration:
   - Nifi installation directory: Directory where Nifi will be installed. (eg. /opt/iemnifi)

2.4.2.2 Installation of Nifi Cluster

Below binaries are used during the installation.

- Nifi Toolkit: https://archive.apache.org/dist/nifi/2.2.0/nifi-toolkit-2.2.0-bin.zip
- Nifi Binary: https://archive.apache.org/dist/nifi/2.2.0/nifi-2.2.0-bin.zip

1. Create a directory for the NiFi installation (e.g., /opt/iemnifi) and navigate into it.

```
sudo mkdir -p /opt/iemnifi
sudo chown -R nifiadmin:nifiadmin /opt/iemnifi
cd /opt/iemnifi
```

2. Download and extract the above Nifi Binary.
3. Run the following commands on all the nodes:

```
wget https://archive.apache.org/dist/nifi/2.2.0/nifi-2.2.0-bin.zip
unzip nifi-2.2.0-bin.zip
```

4. To eliminate version dependency, rename the unzipped folder to nifi.

```
mv nifi-2.2.0 nifi
```

5. Download Nifi-toolkit only on primary server. Use this toolkit link and extract the package on all the nodes. To download Nifi-toolkit, run the following commands:

```
wget https://archive.apache.org/dist/nifi/1.23.2/nifi-toolkit-2.2.0-bin.zip
unzip nifi-toolkit-2.2.0-bin.zip
```

6. To eliminate version dependency, rename the unzipped folder to nifi.

```
mv nifi-toolkit-2.2.0 nifi-toolkit
```

```
total 789560
drwxr-xr-x.  6 nifiadmin nifiadmin        86 Jan 24 15:15 nifi-toolkit
drwxr-xr-x.  3 nifiadmin nifiadmin       120 Jun  6 13:30 data
-rw-r--r--.  1 nifiadmin nifiadmin 729771289 Jun  6 13:35 nifi-2.2.0-bin.zip
-rw-r--r--.  1 nifiadmin nifiadmin  24444539 Jun  6 13:35 nifi-toolkit-2.2.0-bin.zip
-rw-r--r--.  1 nifiadmin nifiadmin  54282240 Jun  6 13:38 data.tar
drwxr-xr-x.  2 nifiadmin nifiadmin        79 Jun  6 13:50 sslcerts
drwxr-xr-x.  3 nifiadmin nifiadmin      4096 Jun  6 14:34 script
drwxrwxr-x. 17 nifiadmin nifiadmin      4096 Jun  6 15:21 nifi
```

7. Create an sslcerts directory on the Primary Server. Execute the following command on the Primary node:

```
mkdir -p sslcerts
```

8. For generating the SSL certificate, run the following command in sslcerts directory:

   Assume the following server details:

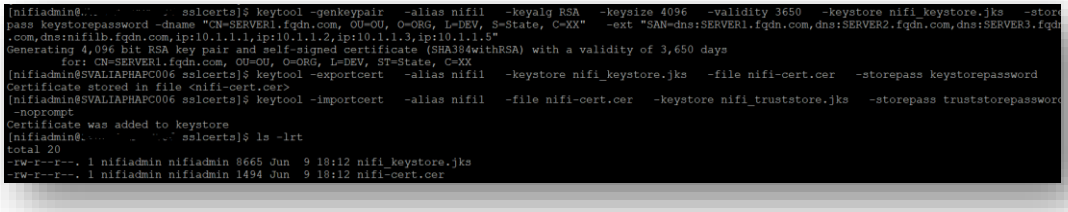
   - Server 1: SERVER1.domain.com – 10.1.1.1

```
cd sslcerts/

keytool -genkeypair   -alias nifi   -keyalg RSA   -keysize 4096
-validity 3650   -keystore nifi_keystore.jks   -storepass
<<keystorepassword>> -dname "CN=SERVER1.domain.com, OU=OU,
O=ORG, L=DEV, S=State, C=XX"   -ext
"SAN=dns:SERVER1.domain.com,ip:10.1.1.1"

keytool -exportcert   -alias nifi   -keystore nifi_keystore.jks
-file nifi-cert.cer   -storepass <<keystorepassword>>
```

```
keytool -importcert   -alias nifi   -file nifi-cert.cer   -
keystore nifi_truststore.jks   -storepass
<<truststorepassword>>   -noprompt
```



9. Copy the appropriate certs to conf directory. Run the following commands on the Primary Node:
   - For copying to primary server:

```
cp sslcerts/* /opt/iemnifi/nifi/conf/
```

10. Change the directory to nifi conf directory.

```
cd /opt/iemnifi/nifi/conf/
```

11. Take Bakup of original configuration file.

```
cp /opt/iemnifi/nifi/conf/nifi.properties
/opt/iemnifi/nifi/conf/nifi.properties.orig
```

12. Set the configuration below in nifi.properties, which were generated from ssl certificate on all the nodes.

```
vim /opt/iemnifi/nifi/conf/nifi.properties
```

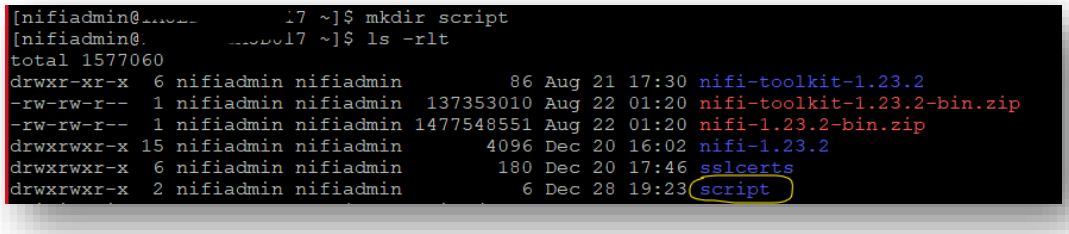
```
nifi.web.https.host=<<server hostname>>
nifi.web.https.port=9443
nifi.remote.input.host=<<server hostname>>
nifi.remote.input.secure=true
nifi.remote.input.socket.port=10443
nifi.security.keystore=./conf/nifi_keystore.jks
nifi.security.keystoreType=JKS
nifi.security.keystorePasswd=keystorepassword
nifi.security.keyPasswd=keystorepassword
nifi.security.truststore=./conf/nifi_truststore.jks
nifi.security.truststoreType=JKS
nifi.security.truststorePasswd=truststorepassword
```

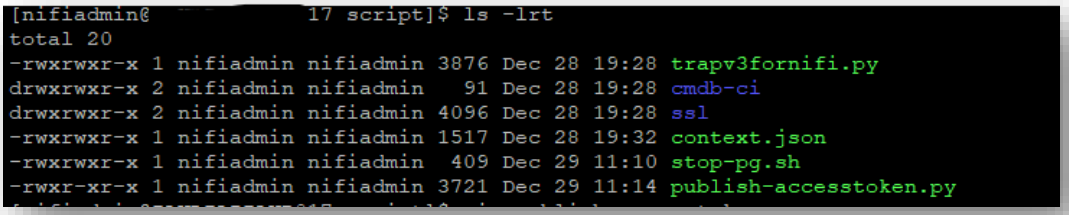
13. Create a script folder to keep all script to be used further. Run the following command:

```
mkdir -p /opt/iemnifi/script/
```

14. Copy the following files and directories to the script folder which was created in the previous step:
    - rwxrwxr-x 1 nifiadmin nifiadmin 3876 Dec 28 19:28 trapv3fornifi.py
    - rwxrwxr-x 1 nifiadmin nifiadmin 425 Dec 28 19:28 stop-pg.sh
    - drwxrwxr-x 2 nifiadmin nifiadmin 91 Dec 28 19:28 cmdb-ci
    - drwxrwxr-x 2 nifiadmin nifiadmin 4096 Dec 28 19:28 ssl

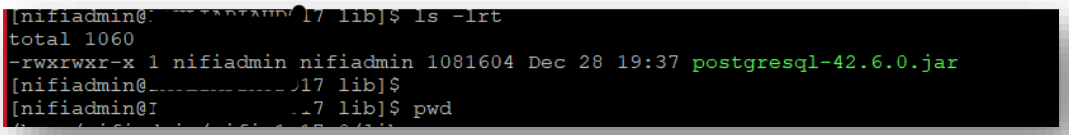
- rwxrwxr-x 1 nifiadmin nifiadmin 1517 Dec 28 19:32 context.json
- rwxr-xr-x 1 nifiadmin nifiadmin 3721 Dec 28 19:33 publish-accesstoken.py





15. Copy Postgres jar file for DB connection in nifi lib directory.

```
cp postgresql-42.6.0.jar /opt/iemnifi/nifi/lib/
```



16. Set credentials by running the following command:

Password length must be 14 characters:

```
sh /opt/iemnifi/nifi/bin/nifi.sh set-single-user-credentials
<username> <password@14letter>
```

17. To make Nifi a Service, edit the bootstrap file:

```
vim /opt/iemnifi/nifi/conf/bootstrap.conf
```

18. Run the command below on all the nodes to modify the config file.

```
run.as: nifiadmin
java.arg.2=-Xms2g
java.arg.3=-Xmx8g
```

```
# Java command to use when running NiFi
java=java

# Username to use when running NiFi. This value will be ignored on Windows.
run.as=nifiadmin

# Preserve shell environment while runnning as "run.as" user
preserve.environment=false
```

19. Start nifi service and check the status on all the nodes:

```
sh /opt/iemnifi/nifi/bin/nifi.sh start
sh /opt/iemnifi/nifi/bin/nifi.sh status
```

```
JAVA_HOME=/usr/lib/jvm/java-21-openjdk-21.0.7.0.6-1.el9.x86_64
NIFI_HOME=/opt/nifi/nifi

2025-06-09 19:06:21,186 INFO [main] org.apache.nifi.bootstrap.Command Application Process [115554] Command Status [SUCCESS] HTTP 200
2025-06-09 19:06:21,188 INFO [main] org.apache.nifi.bootstrap.Command Status: UP
```

20. Once the nifi service is started, application UI becomes accessible on web browser.

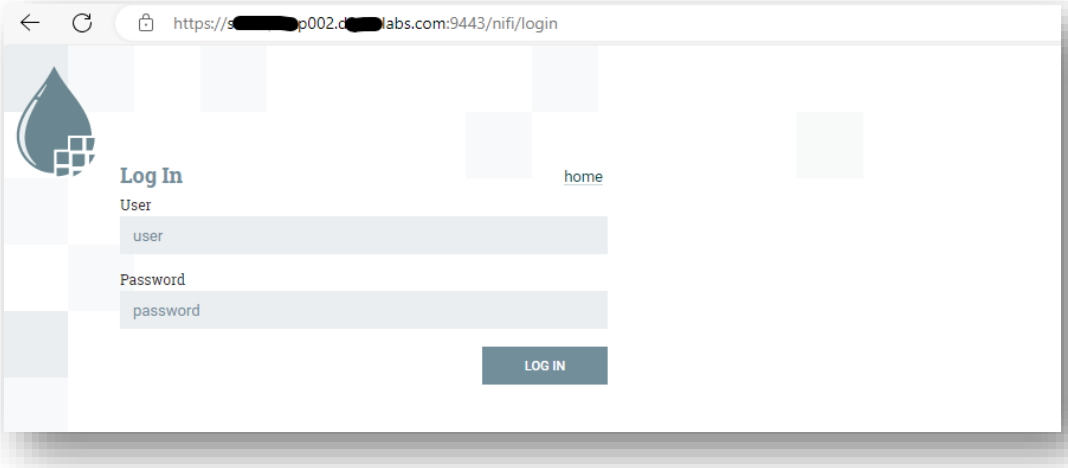


Figure 3. LOG -IN

2.4.3 NiFi Collector Cluster Installation - Manual

2.4.3.1 Pre-requisites for Cluster Installation

Before using Apache NiFi, the following things must be done on your system:

1. A service account with identical credentials must be created on all the specified servers. The username should be consistent (e.g., nifiadmin), and the same credentials should be used across all servers.
2. The service account must be granted sudo privileges on all servers.

Going forward, for the SNMPtrap integration, use the default UDP port 162. It is necessary to make that port unprivileged.

3. Run below command (on all the nodes) using sudo for snmptrap port permission:

```
sysctl -w net. ipv4.ip_unprivileged_port_start=162
```

```
[root@S        P002 ~]# sysctl -w net.ipv4.ip_unprivileged_port_start=162
net.ipv4.ip_unprivileged_port_start = 162
[root@S        P002 ~]#
```

4. Switch to “service account” user to compete the installation steps.
5. Run the below command (on all nodes) for modifying a user account:
6. Check if Java 21 is installed on servers. If not, run the command below (on all the nodes) for JDK installation:

```
sudo yum install java-21-openjdk
```

7. Validate the version of the java. Run the below command (on all nodes) check installed java version:

```
java -version
```

```
openjdk version "21.0.7" 2025-04-15 LTS
OpenJDK Runtime Environment (Red_Hat-21.0.7.0.6-1) (build 21.0.7+6-LTS)
OpenJDK 64-Bit Server VM (Red_Hat-21.0.7.0.6-1) (build 21.0.7+6-LTS, mixed mode, sharing)
```

8. Ensure that the hostname of each server can be resolved from all other servers, and that all servers are mutually reachable over the network.
9. Ensure that the following ports are open between all servers:
   - Nifi.remote.input.socket.port = 10443
   - Nifi.web.https.port = 9443
   - Nifi.cluster.node.protocol.port = 11443
   - Nifi.cluster.load.balance.port = 6342
   - nifi.zookeeper.connect.string=2181, 2888, 3888
10. Create directories and files required for integration:

    Nifi installation directory: Directory where Nifi will be installed. (eg. /opt/iemnifi)

2.4.3.2 Installation of Nifi Cluster

Below binaries are used during the installation of clusters.

- NiFi Toolkit: https://archive.apache.org/dist/nifi/2.2.0/nifi-toolkit-2.2.0-bin.zip
- NiFi Binary: https://archive.apache.org/dist/nifi/2.2.0/nifi-2.2.0-bin.zip

– Create a directory for the NiFi installation (e.g., /opt/iemnifi) and navigate into it.

```
mkdir -p /opt/iemnifi
cd /opt/iemnifi
```

– Download and extract the above Nifi Binary on all servers.

– Run the following commands on all the nodes:

```
wget https://archive.apache.org/dist/nifi/2.2.0/nifi-2.2.0-
bin.zip

unzip nifi-2.2.0-bin.zip
```

– To eliminate version dependency, rename the unzipped folder to NiFi.

```
mv nifi-2.2.0 nifi
```

- Download NiFi-toolkit only on the primary server. Use this toolkit link and extract the package on all the nodes. To download NiFi-toolkit, run the following commands:

```
wget https://archive.apache.org/dist/nifi/1.23.2/nifi-toolkit-2.2.0-bin.zip
unzip nifi-toolkit-2.2.0-bin.zip
```

- To eliminate version dependency, rename the unzipped folder to nifi.

```
mv nifi-toolkit-2.2.0 nifi-toolkit
```

```
total 789560
drwxr-xr-x.  6 nifiadmin nifiadmin        86 Jan 24 15:15 nifi-toolkit
drwxr-xr-x.  3 nifiadmin nifiadmin       120 Jun  6 13:30 data
-rw-r--r--.  1 nifiadmin nifiadmin 729771289 Jun  6 13:35 nifi-2.2.0-bin.zip
-rw-r--r--.  1 nifiadmin nifiadmin  24444539 Jun  6 13:35 nifi-toolkit-2.2.0-bin.zip
-rw-r--r--.  1 nifiadmin nifiadmin  54282240 Jun  6 13:38 data.tar
drwxr-xr-x.  2 nifiadmin nifiadmin        79 Jun  6 13:50 sslcerts
drwxr-xr-x.  3 nifiadmin nifiadmin      4096 Jun  6 14:34 script
drwxrwxr-x. 17 nifiadmin nifiadmin      4096 Jun  6 15:21 nifi
```

- Create an sslcerts directory on the Primary Server. Execute the following command on the Primary node:

```
mkdir -p /opt/iemnifi/sslcerts
```

- For generating the SSL certificate, run the following command in sslcerts directory on the Primary node:

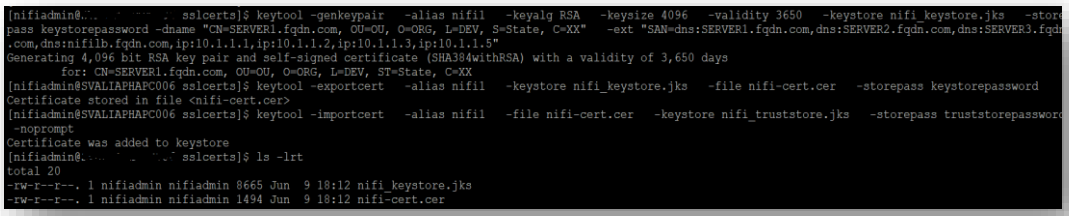
  Assume the following server details:

  - Server 1: SERVER1.domain.com – 10.1.1.1
  - Server 2: SERVER2.domain.com – 10.1.1.2
  - Server 3: SERVER3.domain.com – 10.1.1.3
  - Load Balancer: nifilb.domain.com – 10.1.1.5

```
cd sslcerts/
Need to change the san name
keytool -genkeypair   -alias nifi   -keyalg RSA   -keysize 4096   -validity 3650   -keystore nifi_keystore.jks   -storepass <<keystorepassword>> -dname "CN=SERVER1.domain.com, OU=OU, O=ORG, L=DEV, S=State, C=XX"   -ext "SAN=dns:SERVER1.domain.com,dns:SERVER2.domain.com,dns:SERVER3.domain.com,dns:nifilb.domain.com,ip:10.1.1.1,ip:10.1.1.2,ip:10.1.1.3,ip:10.1.1.5"

keytool -exportcert   -alias nifi   -keystore nifi_keystore.jks   -file nifi-cert.cer   -storepass <<keystorepassword>>

keytool -importcert   -alias nifi   -file nifi-cert.cer   -keystore nifi_truststore.jks   -storepass <<truststorepassword>>   -noprompt
```



- Copy the appropriate sslcert folder to other servers. Run the following commands on the Primary Node:
  - For copying to primary server:

```
cp sslcerts/* /opt/iemnifi/nifi/conf/
```

  - For copying to other servers:

```
scp -r sslcerts SERVER2.domain.com:/opt/iemnifi/
```

```
scp -r sslcerts SERVER3.domain.com:/opt/iemnifi/
```

  - On SERVER2 and SERVER3, add these certificates in nifi conf directory.

```
cp sslcerts/* /opt/iemnifi/nifi/conf/
```

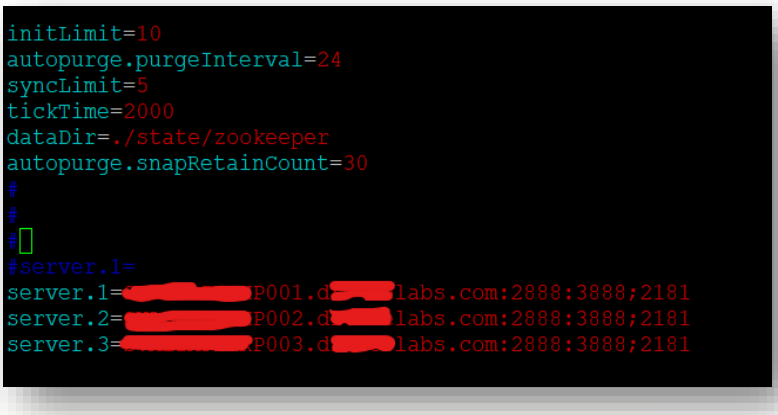
- Change the directory to nifi conf directory (on all the nodes).

```
cd /opt/iemnifi/nifi/conf/
```

- Take backup and edit in zookeeper.properties as per the following table (below lines to be updated): (need to do on all the nodes

```
cp /opt/iemnifi/nifi/conf/zookeeper.properties
/opt/iemnifi/nifi/conf/zookeeper.properties.bkp
vim /opt/iemnifi/nifi/conf/zookeeper.properties
```

```
autopurge.snapRetainCount=30
server.1=Server001.domain.com:2888:3888;2181
server.2=Server002.domain.com:2888:3888;2181
server.3=Server003.domain.com:2888:3888;2181
```



- To create a /state/zookeeper directory, run the below commands (on all the nodes) on the path /opt/iemnifi/nifi.

```
cd /opt/iemnifi/nifi/
```

```
mkdir -p /opt/iemnifi/nifi/state/zookeeper
```

- As mentioned above on the zookeeper.properties, create a “myid” file and set the values of all the nodes respectively to help the cluster to connect accordingly.

  For example: Set the value in myid file as 1 in SERVER1, 2 in SERVER2 and 3 in SERVER3.

- Run the following command to set the values on all the nodes:

```
touch /opt/iemnifi/nifi/state/zookeeper/myid
```

- Update the file and write (1,2,3… respectively as mentioned in zookeeper properties by running the following command:

```
vim /opt/iemnifi/nifi/state/zookeeper/myid
```

- To form a cluster, update the state-management.xml file on all servers as mentioned below.

```
vim /opt/iemnifi/nifi/conf/state-management.xml
```

- Search for “cluster-provider” keyword and only the line marked in bold need to be updated.
- **Sample View**:

```
<cluster-provider>
        <id>zk-provider</id>
        <class>org. apache. nifi.
controller.state.providers.zookeeper.ZooKeeperStateProvider</cl
ass>
        <property name="Connect
String">SERVER1.domain.com:2181,
SERVER2.domain.com:2181,SERVER3.domain.com:2181</property>
        <property name="Root Node">/nifi</property>
        <property name="Session Timeout">10 seconds</property>
        <property name="Access Control">Open</property>
    </cluster-provider>
```

- Set the configuration below in nifi.properties, (on all the nodes). Only the lines marked in bold need to be updated.

```
vim /opt/iemnifi/nifi/conf/nifi.properties
```

```
nifi.state.management.embedded.zookeeper.start=true
nifi.remote.input.host= SERVER1/2/3.domain.com (seprate on each
server)
nifi.remote.input.secure=true
nifi.remote.input.socket.port=10443
nifi.remote.input.http.enabled=true
nifi.web.https.host=Server1/2/3.domain.com (Server FQDN)
nifi.web.https.port=9443
nifi.web. proxy.host=localhost:9443,Server1/2/3.domain.com:9443
(Server FQDN), LB.domain.com:9443
nifi.sensitive.props.key=propkeywith12chars
nifi.cluster.protocol.is.secure=true
```

```
nifi.cluster.is.node=true
nifi.cluster.node.address= Server001/002/003.domain.com
nifi.cluster.node.protocol.port=11443
nifi.cluster.load.balance.host= Server1/2/3.domain.com (Server
FQDN) nifi.cluster.load.balance.port=6342
nifi.zookeeper.connect.string=SERVER1.domain.com:2181,
SERVER2.domain.com:2181, SERVER3.domain.com:2181
nifi.security.keystore=./conf/nifi_keystore.jks
nifi.security.keystore.certificate=
nifi.security.keystore.privateKey=
nifi.security.keystoreType=JKS
nifi.security.keystorePasswd=keystorepassword
nifi.security.keyPasswd=keystorepassword
nifi.security.truststore=./conf/nifi_truststore.jks
nifi.security.truststore.certificate=
nifi.security.truststoreType=JKS
nifi.security.truststorePasswd=truststorepassword
```

- Create a script folder to keep all script to be used further. Run the following command on all the nodes:

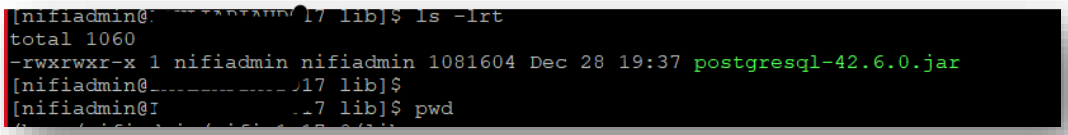
```
mkdir -p /opt/iemnifi/script/
```

- Copy the following files and directories to the script folder which was created in the previous step:
  - rwxrwxr-x 1 nifiadmin nifiadmin 3876 Dec 28 19:28 trapv3fornifi.py
  - rwxrwxr-x 1 nifiadmin nifiadmin 425 Dec 28 19:28 stop-pg.sh
  - drwxrwxr-x 2 nifiadmin nifiadmin 91 Dec 28 19:28 cmdb-ci
  - drwxrwxr-x 2 nifiadmin nifiadmin 4096 Dec 28 19:28 ssl
  - rwxrwxr-x 1 nifiadmin nifiadmin 1517 Dec 28 19:32 context.json
  - rwxr-xr-x 1 nifiadmin nifiadmin 3721 Dec 28 19:33 publish-accesstoken.py

```
[nifiadmin@          17 ~]$ mkdir script
[nifiadmin@          17 ~]$ ls -rlt
total 1577060
drwxr-xr-x  6 nifiadmin nifiadmin         86 Aug 21 17:30 nifi-toolkit-1.23.2
-rw-rw-r--  1 nifiadmin nifiadmin  137353010 Aug 22 01:20 nifi-toolkit-1.23.2-bin.zip
-rw-rw-r--  1 nifiadmin nifiadmin 1477548551 Aug 22 01:20 nifi-1.23.2-bin.zip
drwxrwxr-x 15 nifiadmin nifiadmin       4096 Dec 20 16:02 nifi-1.23.2
drwxrwxr-x  6 nifiadmin nifiadmin        180 Dec 20 17:46 sslcerts
drwxrwxr-x  2 nifiadmin nifiadmin          6 Dec 28 19:23 script
```

```
[nifiadmin@          17 script]$ ls -lrt
total 20
-rwxrwxr-x 1 nifiadmin nifiadmin 3876 Dec 28 19:28 trapv3fornifi.py
drwxrwxr-x 2 nifiadmin nifiadmin   91 Dec 28 19:28 cmdb-ci
drwxrwxr-x 2 nifiadmin nifiadmin 4096 Dec 28 19:28 ssl
-rwxrwxr-x 1 nifiadmin nifiadmin 1517 Dec 28 19:32 context.json
-rwxrwxr-x 1 nifiadmin nifiadmin  409 Dec 29 11:10 stop-pg.sh
-rwxr-xr-x 1 nifiadmin nifiadmin 3721 Dec 29 11:14 publish-accesstoken.py
```

- Copy Postgres jar file for DB connection in nifi lib directory on all nodes.



- Set the same credentials on all the nodes by running the following command:

Password length must be 14 characters:

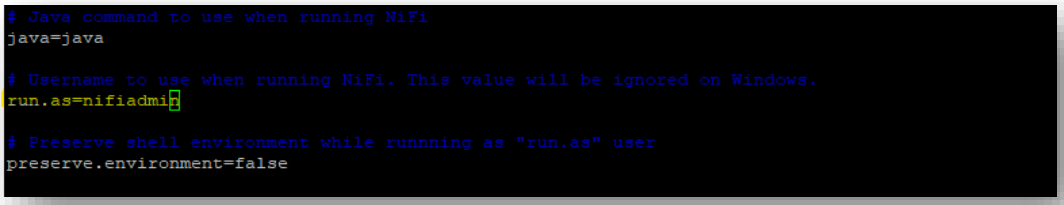
```
sh /opt/iemnifi/nifi/bin/nifi.sh set-single-user-credentials
<username> <password@14letter>
```

- To make Nifi a Service, edit the bootstrap file:

```
vim /opt/iemnifi/nifi/conf/bootstrap.conf
```

- Run the command below on all the nodes to modify the config file.

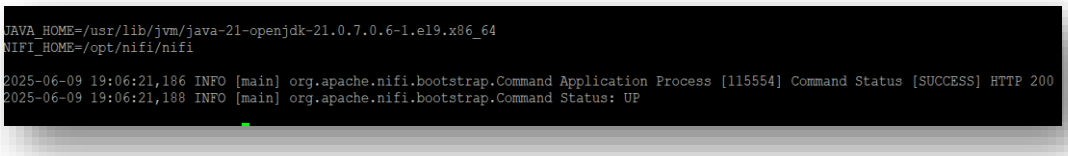
```
run.as: nifiadmin
```



- Start nifi service and check the status on all the nodes:

```
sh /opt/iemnifi/nifi/bin/nifi.sh start
```

```
sh /opt/iemnifi/nifi/bin/nifi.sh status
```



- Once the nifi service is started, application UI becomes accessible on web browser.

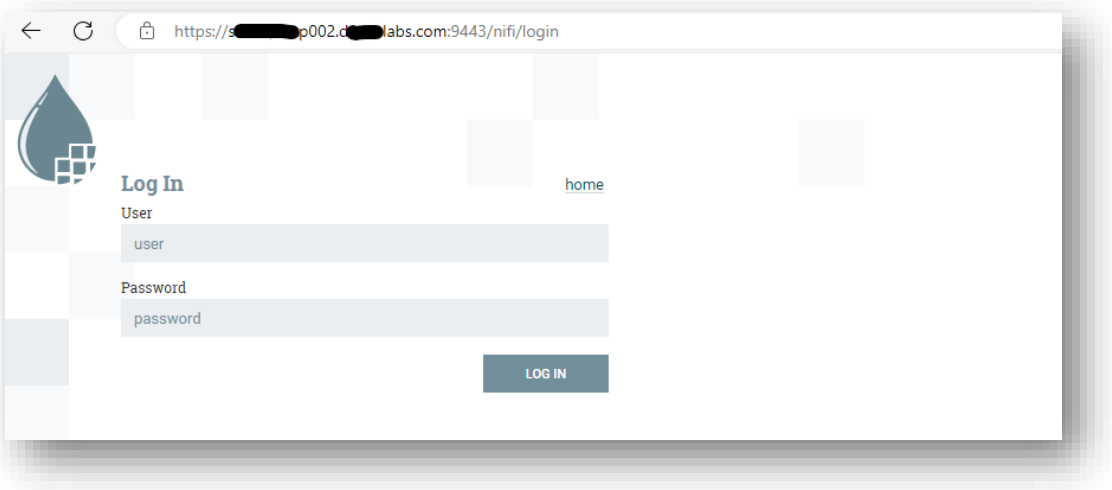


Figure 4 – Log in

2.5 Overview of IMM

Integration Management Module (IMM) is a component of IntelliOps Event Management which is used for 3rd party tools integration and ingesting events, metric, performance, and configuration data into IntelliOps Event Management for performing event management functions.

Using IMM, we can reduce the implementation timeline significantly, allowing you to quickly get the NiFi connectors onboard and take control of the event management ecosystem.

2.6 Prerequisite for IMM

Prerequisites are specific conditions that need to be met before initiating the configuration. Hence, mentioned below are pre-requisites for IMM:

2.6.1 Supported OS for IMM

- Linux RHEL 8.x

2.6.2 Supported DB for IMM

- PostgreSQL 15

2.6.3 Supported Web Browsers

- Microsoft Edge: Current or previous version
- Mozilla FireFox: Current or previous version
- Google Chrome: Current or previous version
- Safari: Current or previous version

2.6.4 Hardware Sizing Recommendation

- 2 Web servers & 2 DB servers are required with below configuration:
  - Web Server: 2CPU, 4GB
  - DB Server: 4CPU, 8GB

2.6.5 Port Requirement for IMM

- IMM KRS Service -4000
- IMM API Service - 4100
- IMM Web Portal - 4200
- IMM Orchestrator Service – 4300

2.7 PostgreSQL

PostgreSQL, also known as Postgres, is a powerful open-source relational database management system (RDBMS) that can run on various operating systems, including Windows, Linux, macOS, FreeBSD, etc. It is known for its reliability, stability, and security. It is the world's most advanced open-source database.

This open-source and object-oriented platform allows you to work with both non-relational & relational queries by leveraging the JSON (JavaScript Object Notation) format.

PostgreSQL is capable of handling large amounts of data, complex transactions, and multi-user environments, making it suitable for various purposes such as web development, data warehousing, and business intelligence.

Table 4 – PostgreSQL Requirements

Version	15
Purpose	It is an open-source relational and non-relational database System
Source	https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-x86_64/pgdg-redhat-repo-latest.noarch.rpm

2.7.1 PostgreSQL Standalone Installation

2.7.1.1 Installation Steps

1. Installing PostgreSQL Packages:
   a. Open a Terminal Window
   b. Find the version of PostgreSQL to install on RHEL 8

```
# sudo yum module list | grep postgresql
```

   c. PostgreSQL is included in the default repositories of RHEL 8, and can be installed using the following dnf command. It will install the PostgreSQL server 10, libraries and client binaries.

```
#  sudo dnf install -y
https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-
x86_64/pgdg-redhat-repo-latest.noarch.rpm
```

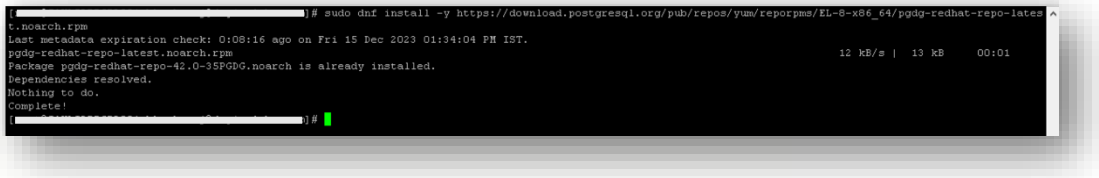


Figure 5 – DNF install

```
# dnf update
```

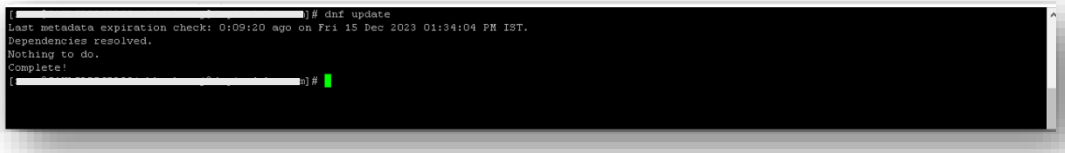


Figure 6 – DNF Update

Note: The default built-in **PostgreSQL** module might lead to unwanted conflicts, make sure it disabled.

```
# SUDO DNF -QY MODULE DISABLE POSTGRESQL
```

d. We can now proceed with the installation of the PostgreSQL15 database server.

```
# dnf install postgresql15-server postgresql15 postgresql15-
contrib
```

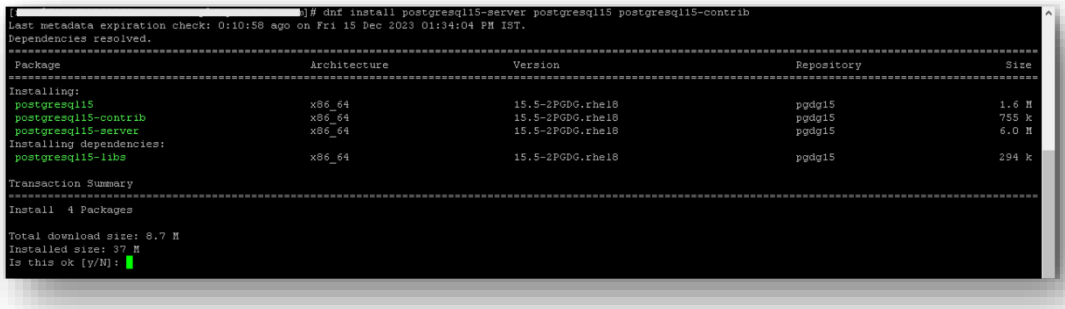


Figure 7 – installing PostgreSQL15

Figure 8– installing PostgreSQL15 (Cont.)

2. Initialize the PostgreSQL Database

a. Once you have installed the PostgreSQL packages, the next step is to initialize the new PostgreSQL database cluster using the /usr/bin/postgresql-setup utility, as follows.

```
# sudo /usr/pgsql-15/bin/postgresql-15-setup initdb
```

Figure 9 - Initialize New PostgreSQL Database Cluster

b. Now that the PostgreSQL cluster is initialized, you need to start the PostgreSQL service, for now, then enable it to auto-start at system boot and verify its status using the systemctl command.

```
# sudo systemctl start postgresql-15
# sudo systemctl enable postgresql-15
# sudo systemctl status postgresql-15
```

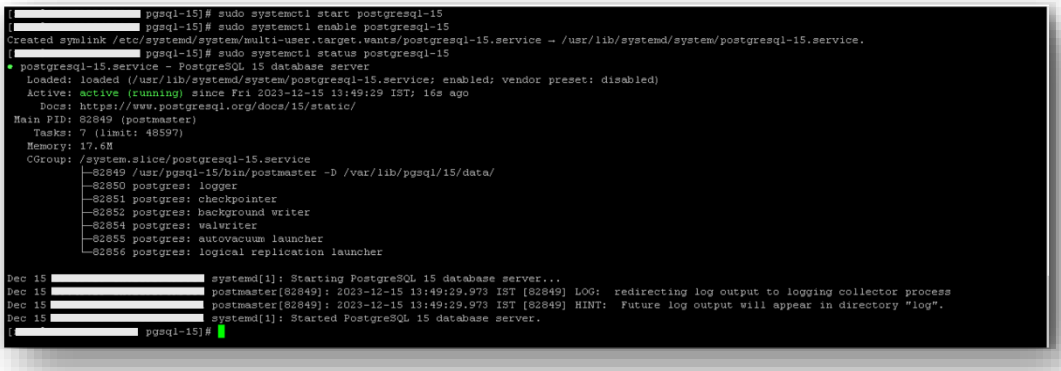


Figure 10 – Systemctl Status

3. Secure and configure PostgreSQL Database

To secure the Postgres user account and the administrative user account.

a. Create a password for a postgres system user account using the passwd utility as follows.

```
# passwd postgres
```

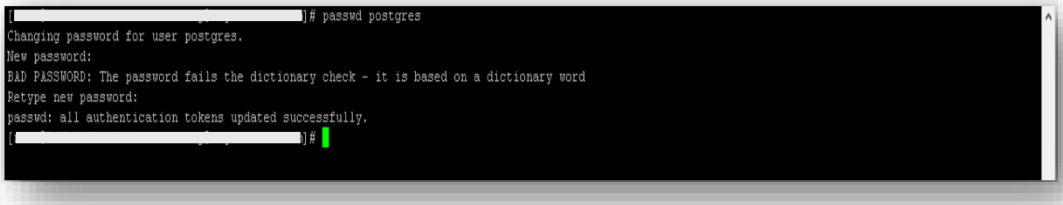


Figure 11 - Create Password

b. Switch to the postgres system user account and secure the PostgreSQL administrative database user account by creating a password for it (remember to set a strong and secure password).

```
# sudo su – postgres
# psql -c "ALTER USER postgres WITH PASSWORD '<Strong
Password>';"
psql -c "ALTER USER postgres WITH PASSWORD '<Password>';"
```

Figure 12 – Alter Role

Figure 13 – Alter Role (Cont.)

c. Create a New PostgreSQL "immdbuser" User Account

i. First create Linux User Account

```
# Sudo useradd immdbuser
# SUDO PASSWD IMMDBUSER
```

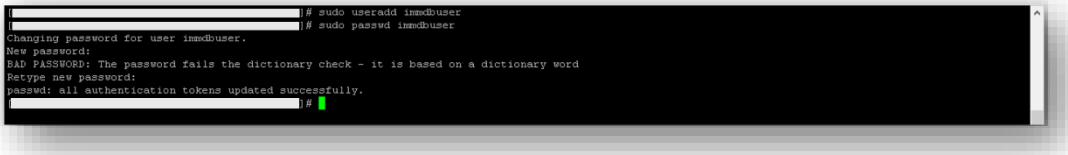


Figure 14 – Create User Account

ii. The postgres account is nothing but an administrative user for PostgreSQL server. So log in as postgres:

```
# sudo -i -u postgres
```

iii. Run the following createuser command to create a new PostgreSQL role for immdbuser Linux user with password (strong and secure password)

```
$ createuser --interactive --pwprompt
```

Figure 15 – Create new PostgreSQL Role

d. The various PostgreSQL configuration files can be found in the /var/lib/pgsql/data/ directory. To view the directory structure, you can use the tree (install it using dnf install tree) command.

```
# tree -L 1 /var/lib/pgsql/15/data/
```

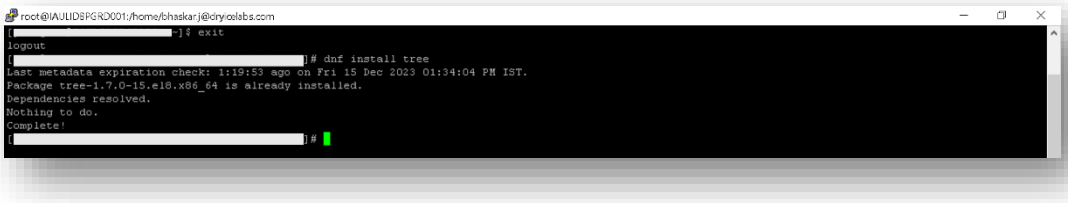


Figure 16 – DNF install Tree

The main server configuration file is /var/lib/pgsql/15/data/postgresql.conf. And the client authentication can be configured using the /var/lib/pgsql/15/data/pg_hba.conf.

e. To enable PostgreSQL remote connection on Redhat, you need to open the following file.

```
# sudo vi /var/lib/pgsql/15/data/postgresql.conf
```

Figure 17 – VI postgresql.conf

f. Uncomment the following parameter available under the 'Connections and Authentication' section.

```
#listen_addresses = 'localhost'
```

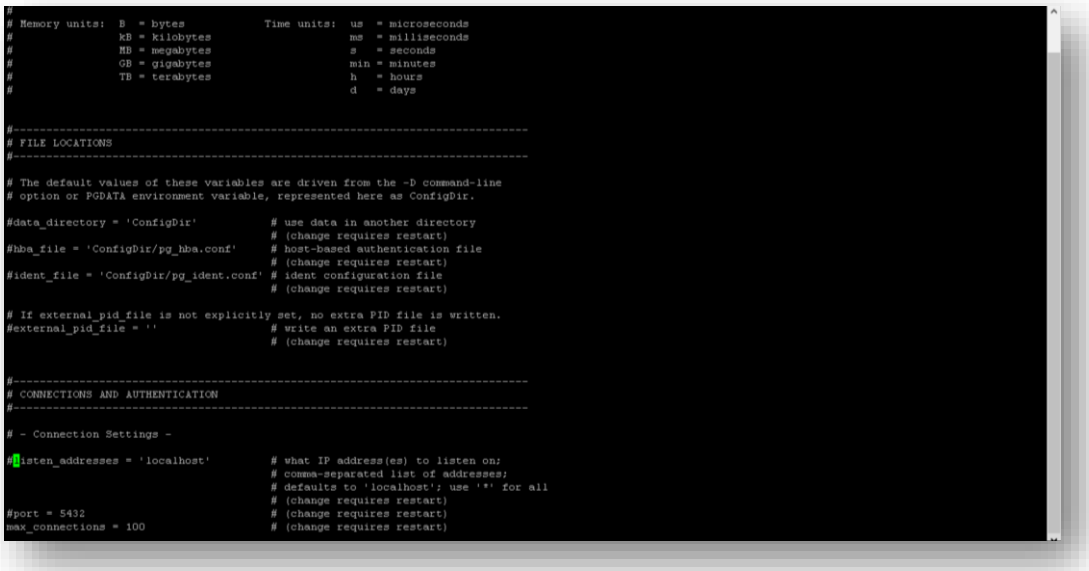


Figure 18 – Connection Setting

g. we are enabling PostgreSQL server connections to accept connections from all IP addresses.

```
listen_addresses = '*'
```

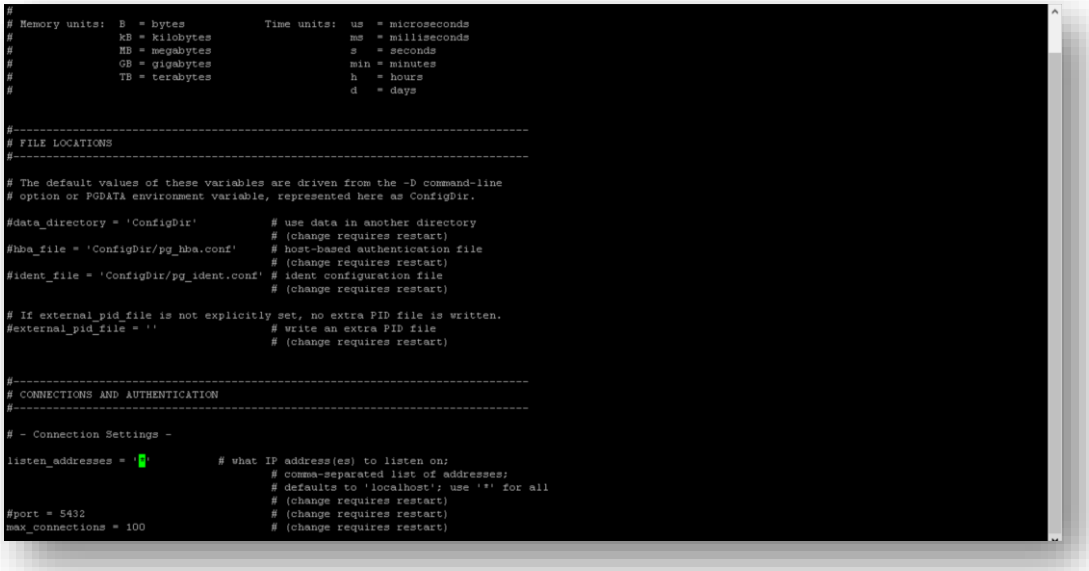


Figure 19 – Connection Setting (Cont.)

h. To apply the change, restart the PostgreSQL service using the following command:

```
# sudo systemctl restart postgresql-15
```

i. PostgreSQL database system supports different types of authentications including password-based authentication. Under the password-based authentication, you can use one of the following methods: md5, crypt, or password (sends the password in clear-text). All are same but major difference between: which way a user's password is stored (on the server) and sent across the connection, when entered by a user. client authentication can be configured using the /var/lib/pgsql/15/data/pg_hba.conf

To prevent password sniffing by attackers and avoid storing passwords on the server in plain text, it is recommended to use **md5** as shown. Now open the client authentication configuration file.

```
# vi /var/lib/pgsql/15/data/pg_hba.conf
```

```
# sudo vi  /var/lib/pgsql/15/data/pg_hba.conf
```

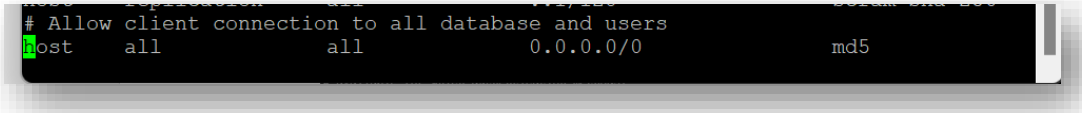


Figure 20 – Security Setting

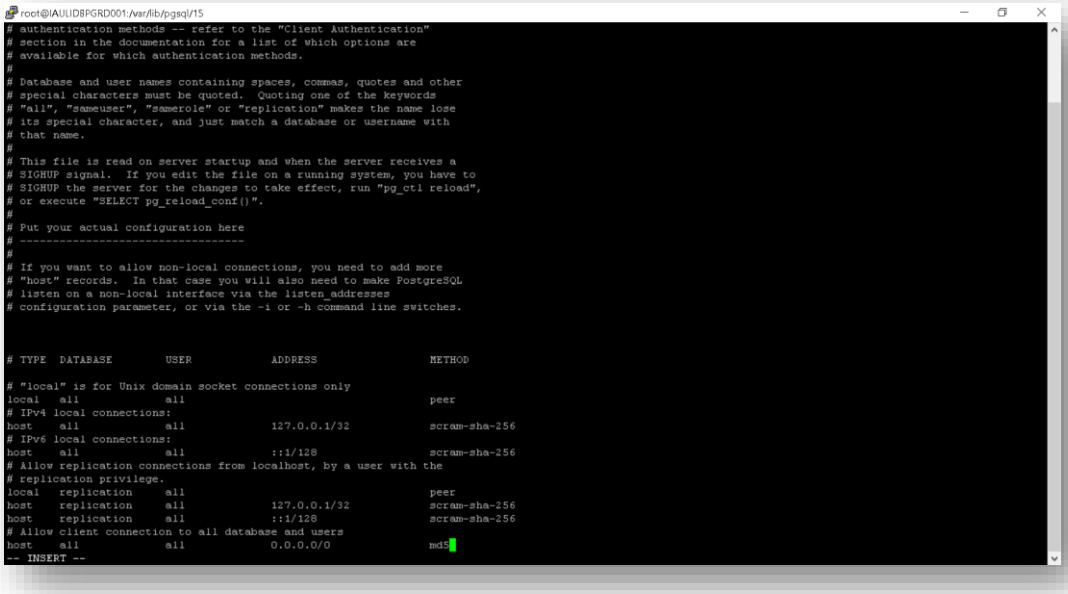


Figure 21 – Security Setting (Cont.)

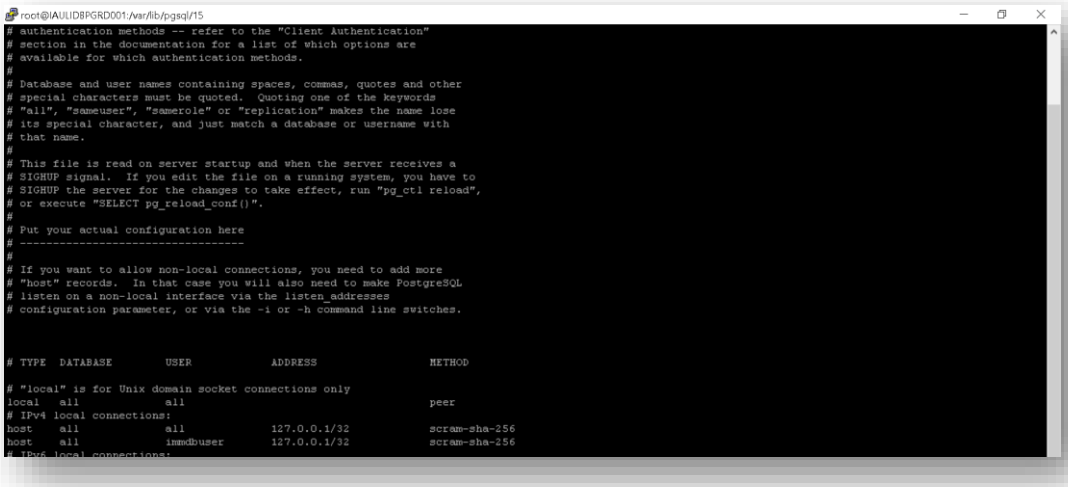


Figure 22 – Security Setting (Cont.)

And look for the following lines and change the authentication method to md5.

And added New User “immdbuser”.

j. Now restart the Postgres service to apply the recent changes in the configuration.

```
# sudo systemctl restart postgresql-15
```

k. Your PostgreSQL database server installation is now secure. You can switch to the postgres account and start working with PostgreSQL.

```
# su - postgres
$ psql
```

2.7.2 PostgreSQL Installation with HA Mode

PostgreSQL maintains the high availability of its clusters by ensuring that a secondary server will take over if the primary server crashes.

2.7.2.1 Environment

- Red Hat Enterprise Linux 8 with High-Availability Add-on running Pacemaker cluster
- PostgreSQL Database Setup

2.7.2.2 Installation Steps

Installing PostgreSQL Packages

1. Open a Terminal Window
2. Find the version of PostgreSQL to install on RHEL 8

```
# sudo yum module list | grep postgresql
```

3. PostgreSQL is included in the default repositories of RHEL 8, and can be installed using the following dnf command, which will install the PostgreSQL server 10, libraries and client binaries.

```
#  sudo dnf install -y
https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-
x86_64/pgdg-redhat-repo-latest.noarch.rpm
```

2.8 IMM Installation and Setup

This section describes the detailed IMM installation procedure, and the various stages involved in this process.

2.8.1 IMM Components

IMM follows multi-tier architecture and includes the following components:

- **Web Components**- This includes the user interface that enables the users to perform various tasks using the IMM Interface.
- **Application Components**- This includes essential services that work together to achieve the core functionality of IMM.

Before starting the installation, it is important to identify the components that the user needs to install based on the requirement. The following table lists the components available on different servers.

Table 5 - Types of Servers

Server Type	Components	Description
Web Component	Web UI	It is the user interface that enables the users to perform various tasks using the IMM Interface
	Web API	It is an API in the IMM web module that can be accessed using the HTTP protocol.
	KRS	The Key Rotation Service component serves the purpose of providing additional security through rotation of keys on a periodic basis.
	Orchestrator	Streamlining and coordinating Module Interactions for seamless execution.
Application Component	Listener	A technical component responsible for actively monitoring and capturing incoming data or events from various sources, enabling real-time processing, and triggering subsequent actions

2.8.2 IMM Installation

This section explains how to install IMM components using the installer on Linux server or standalone machine. The installer can be further used for deployment of web server, application server and , database server.

2.8.2.1 Run the Installer

Review the prerequisites carefully before proceeding with the installation.

After confirming that the system meets the prerequisites to run the IMM installer, perform the following steps:

1. Open putty and Login with valid credentials to the targeted server where you want to install IMM Application.

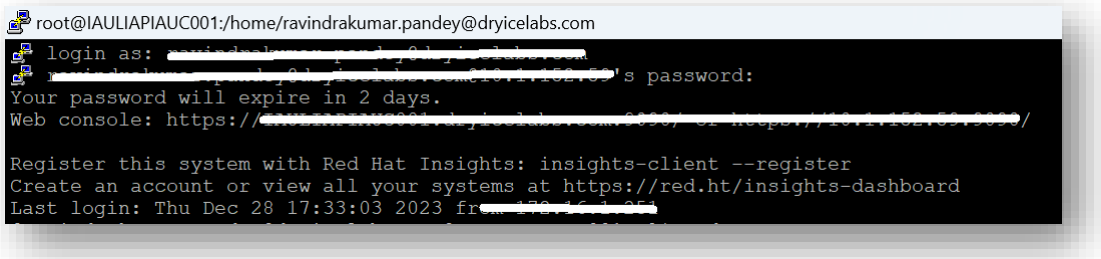


Figure 23 – Login to Server.

2. Run "`sudo su`" command to elevate user to root. Input the user password when prompted.

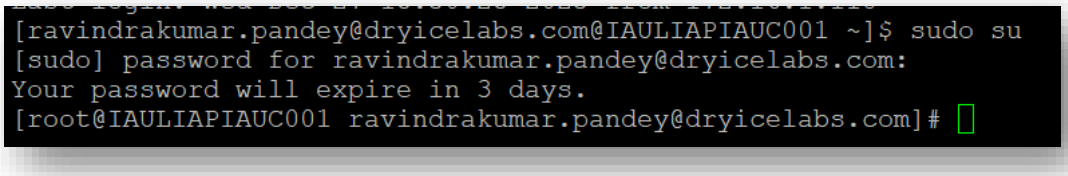


Figure 24 – Elevate the user.

3. Navigate to Installer path where installer file is located by running the following command:

```
cd <installer path>
```

Ex: cd /usr/local/bin/IMMSetup

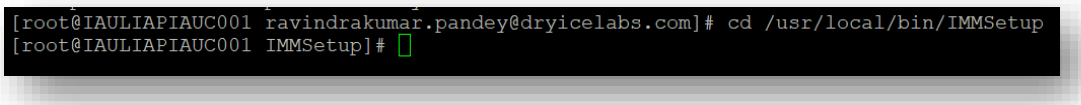


Figure 25 – Navigate to Installer path.

4. Give run access to the IMM installer file.

```
sudo chmod 777 -R <IMM installer path>
```

Ex: sudo chmod 777 -R /usr/local/bin/IMMSetup

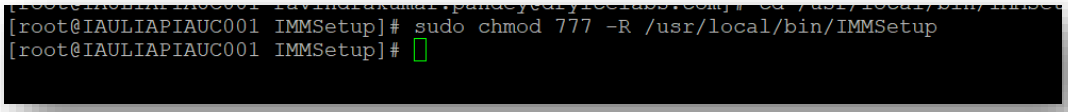


Figure 26 – Provide access to file.

5. Run the installer by typing following command on installer file location.

```
./HCL.IMM.LinuxInstaller
```

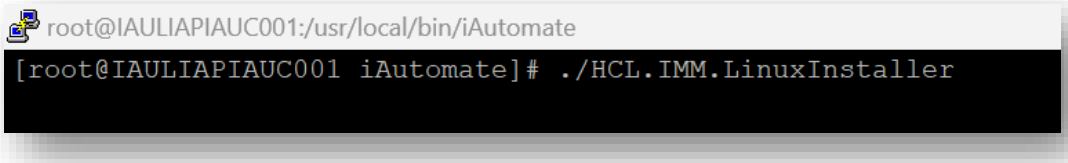


Figure 27 - Run the Installer

2.8.2.2 Install IMM

This section lists the steps to install the IMM components on all Linux servers. Ensure that user meets all requirements in the section Prerequisite for IMM**Error! Reference source not found.** and Table 5 - Types of Servers before starting the installation procedure.

To install IMM, perform the following steps:

1. On running the Installer, the following page appears.

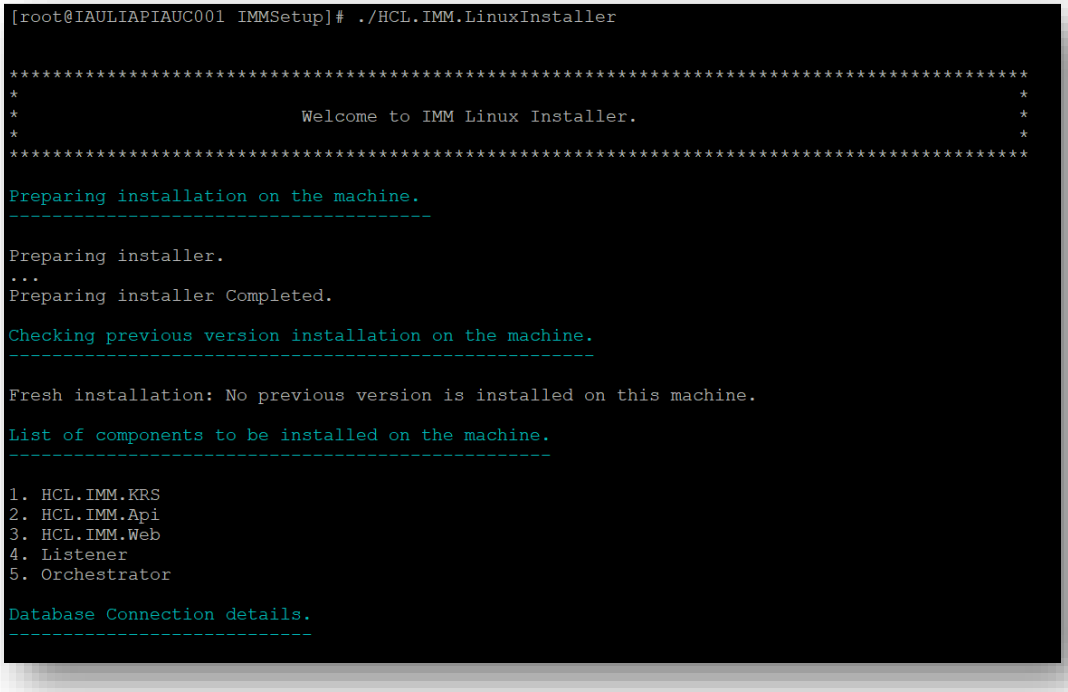


Figure 28 - IMM Installer

2. The installer checks if any previous version is installed on the machine. If not, it runs the fresh installation and lists out all the components that need to be installed on the machine as shown in Figure 28 - IMM Installer.
3. To access database, the installer requires the database connection details. Input your database server host.

   For Validation: Put any value in this field as it has empty validation.

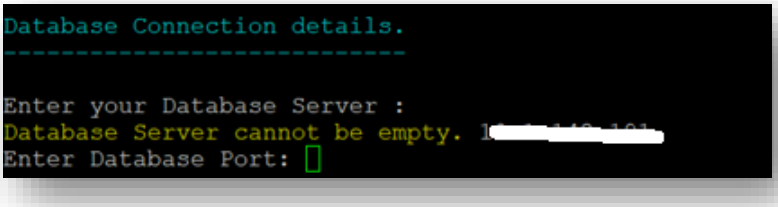


Figure 29 – Database Connection Details, Database Server.

4. Input your **Database Port**. If no port is provided, it uses the default port i.e., 5432.

   For validation: input any valid number in this field as it has number only validation.

The port number must be greater than zero.

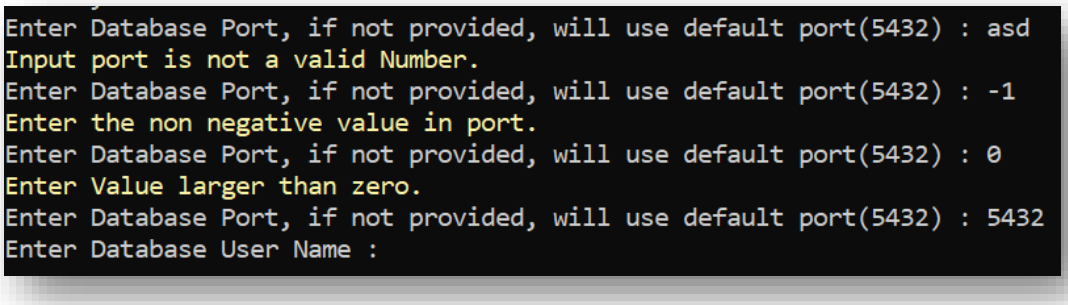


Figure 30 – Database Connection Details, Database Port.

5. Input the **Database Username**.

   For Validation: Put any value in this field as it has empty validation.

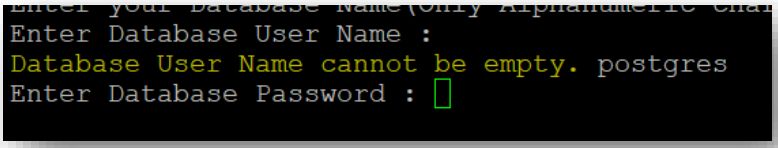


Figure 31 – Database Connection Details, Database Username.

6. Input **Database Password**. Input password will be masked on screen for security purposes.

   For Validation: Put any value in this field as it has empty validation.

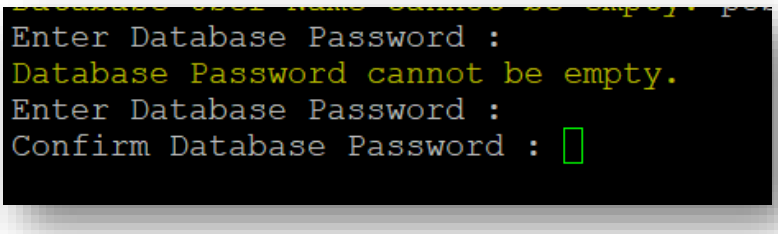


Figure 32 – Database Connection Details, Database password.

7. After getting the connection details, the installer validates the **Database Connection**. In the case of a wrong input, it displays an error message and does not procced further and the user is redirected to the **Input Database Connection Details** stage.

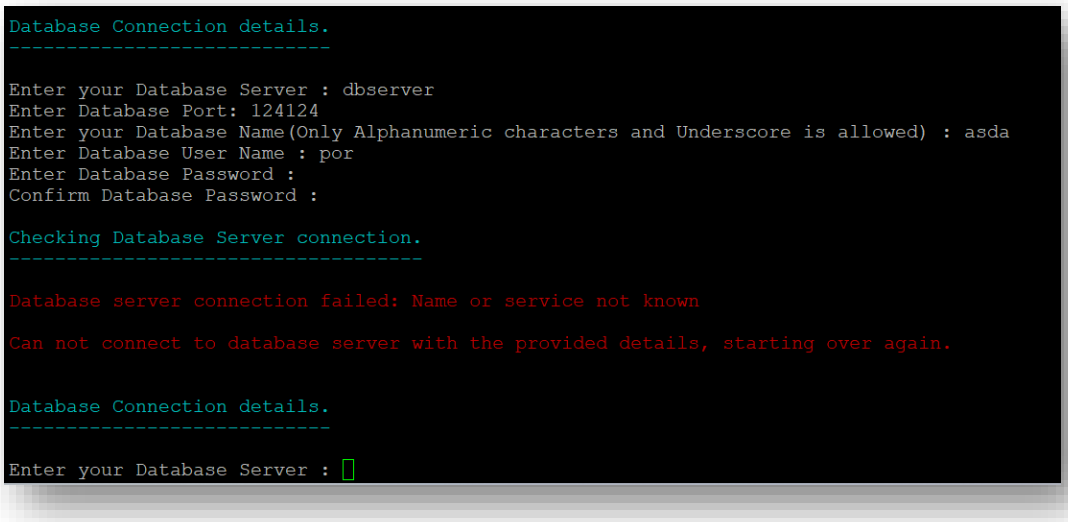


Figure 33 – Database Connection Validation – Connection Failed.

8. It only proceeds after the correct connection details are entered by the user.

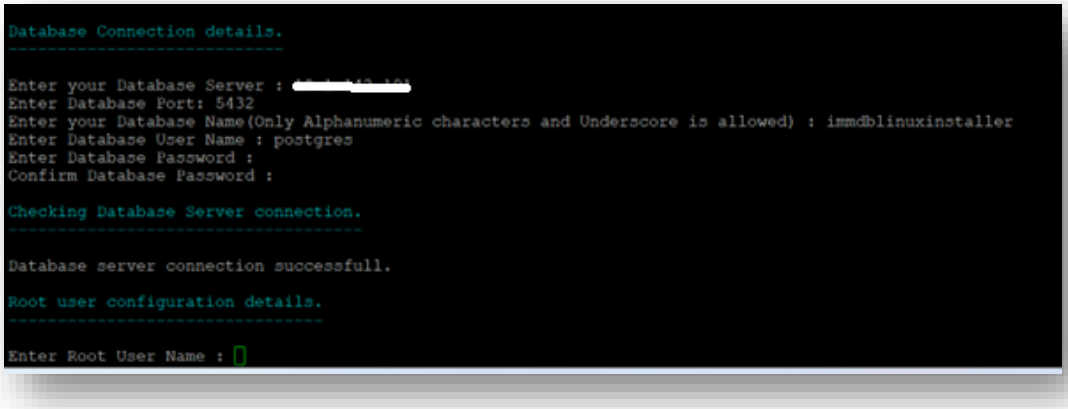


Figure 34 – Database Connection Validation – Connection Succes.

9. Now the Installer captures the Root user configuration details. Enter the **Root Username**.

   For validation: Put any value in this field as it has empty validation.

It has length validation. You can enter root usernames of up to 1000 characters length.

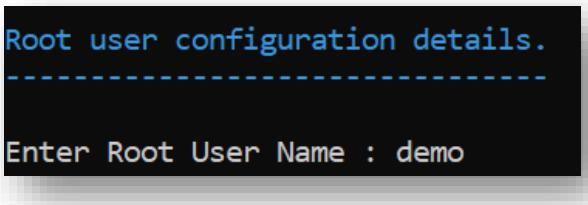


Figure 35 – Root User details – Root Username

10. Enter the **First Name**.

    Validation: Put any value in this field as it has empty validation.

It has length validation. You can enter first name of up to 500 characters length.

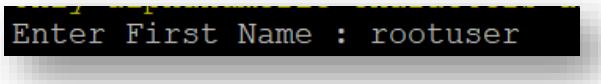


Figure 36 – Root User details – First Name

11. Enter the **Last Name**.
12. Validation: Only Alphanumeric characters and Underscore are allowed in this field.

It has length validation. You can enter first name of up to 500 characters length.

Figure 37 – Root User details – Last Name

13. Enter the **Root User Email**.

    Validation: It has valid email input validation.

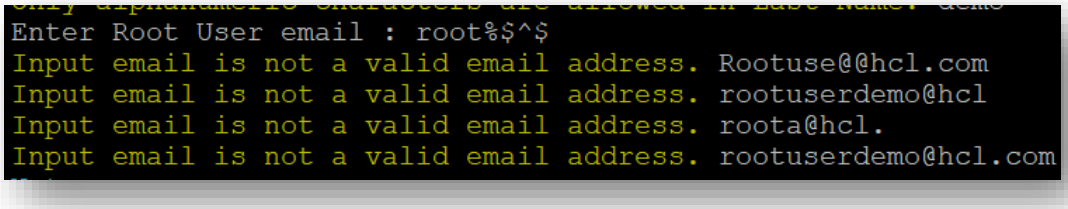


Figure 38 – Root User details – Root User Email

14. Input your **Root User Password**.

    Validation: Put any value in this field as it has empty validation.

Input password is masked on screen for security purposes.

It has Password strength check validation. The input password should follow bellow mentioned rules:

1- Password must contain at least one uppercase letter.

2- Password must contain only these (-.! @#$_^*) special characters.

3- Password must contain at least one number.

4- Password must contain at least one lowercase letter.

5- Password must be of minimum 12 and maximum 500 characters length.

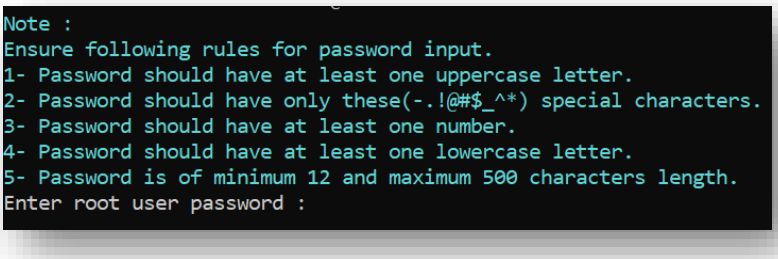


Figure 39 – Root User details – Root User Password.

15. Confirm your root user password. Input password will be masked on screen for security purposes.

    Validation: it has password match validation. The installer will proceed only after the input password and confirm password matches.

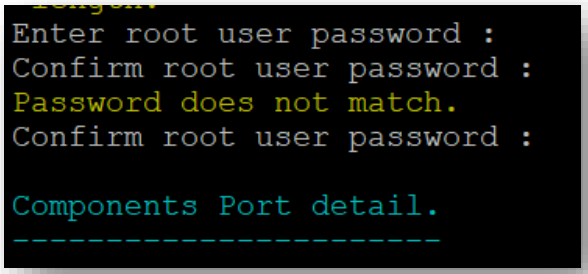


Figure 40 - Root User details – Confirm Root User Password.

16. Now, enter the ports for KRS, API, Web, and Orchestrator applications.

    Validation: The Installer checks for entered port for every application whether they are open - listening and not in use by any other application.

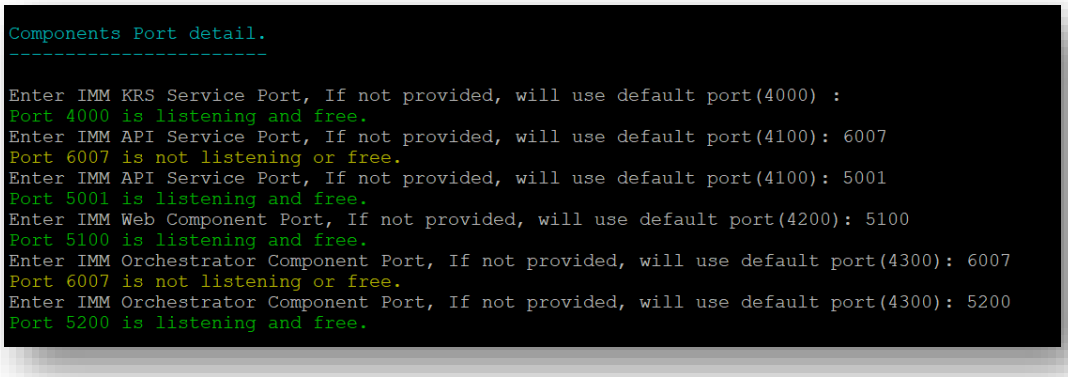


Figure 41 – Input port for KRS, API, Web, and Orchestrator applications

17. After capturing all the details, the installer confirms from the user if he wants to proceed with the installation.

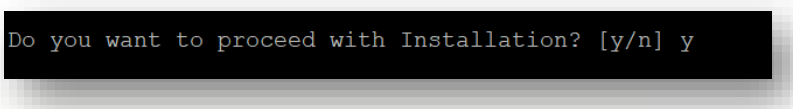


Figure 42 – Installation confirmation before proceeding further.

18. Installer further checks if the input database exists. If not, it creates it and runs the database scripts on it. It also creates the root user and maps this user to root admin role.

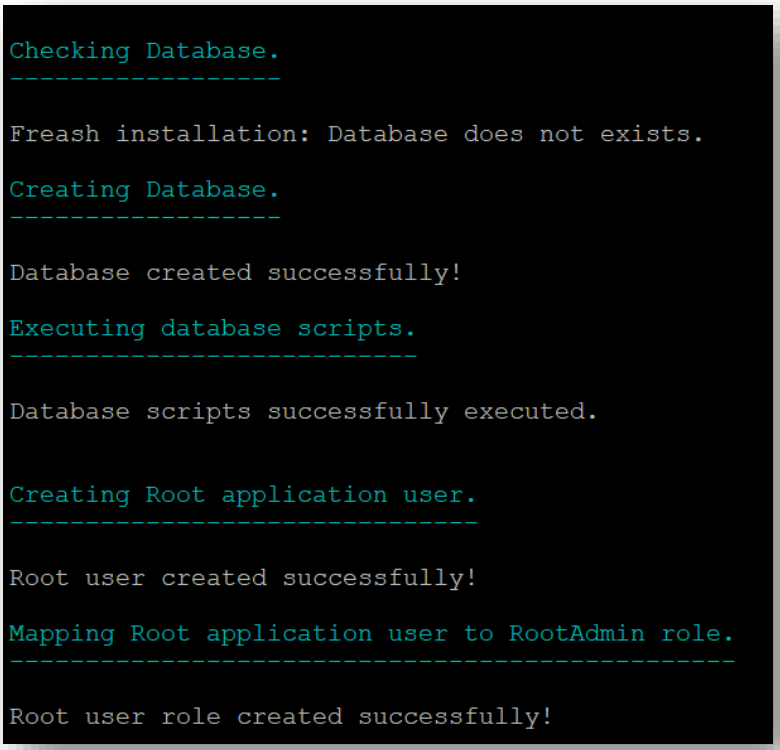


Figure 43 – Installation Database Check and Other Database Tasks

The components installation starts.

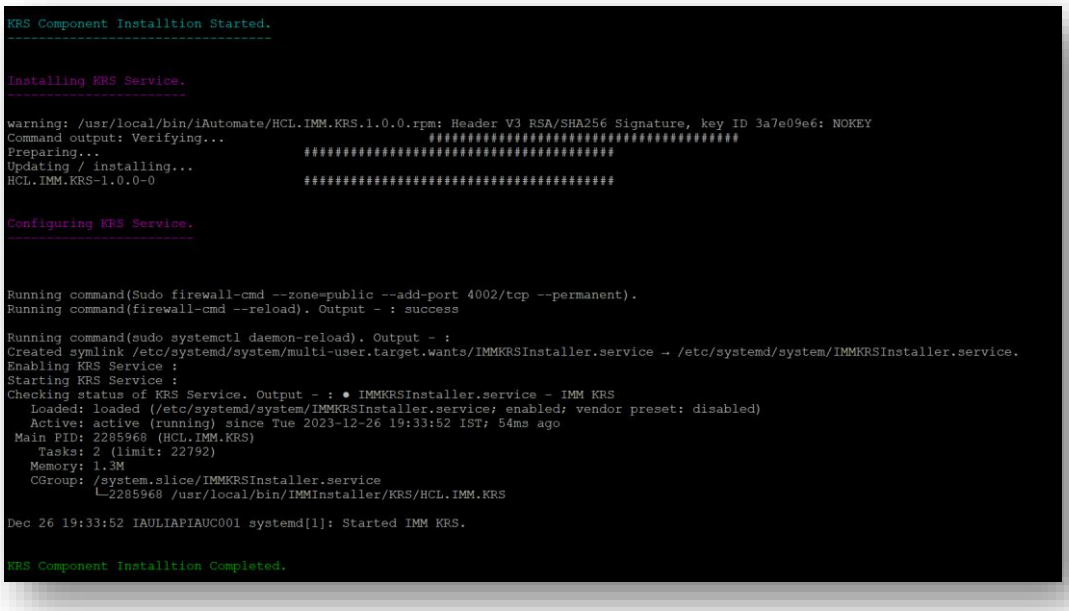


Figure 44 – KRS Component Installation

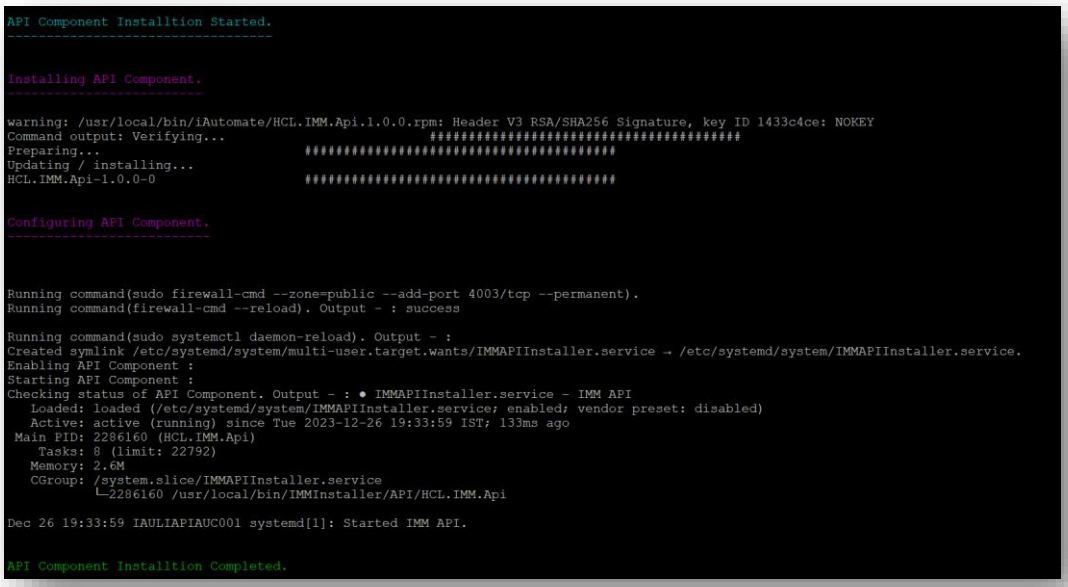


Figure 45 – API Component Installation

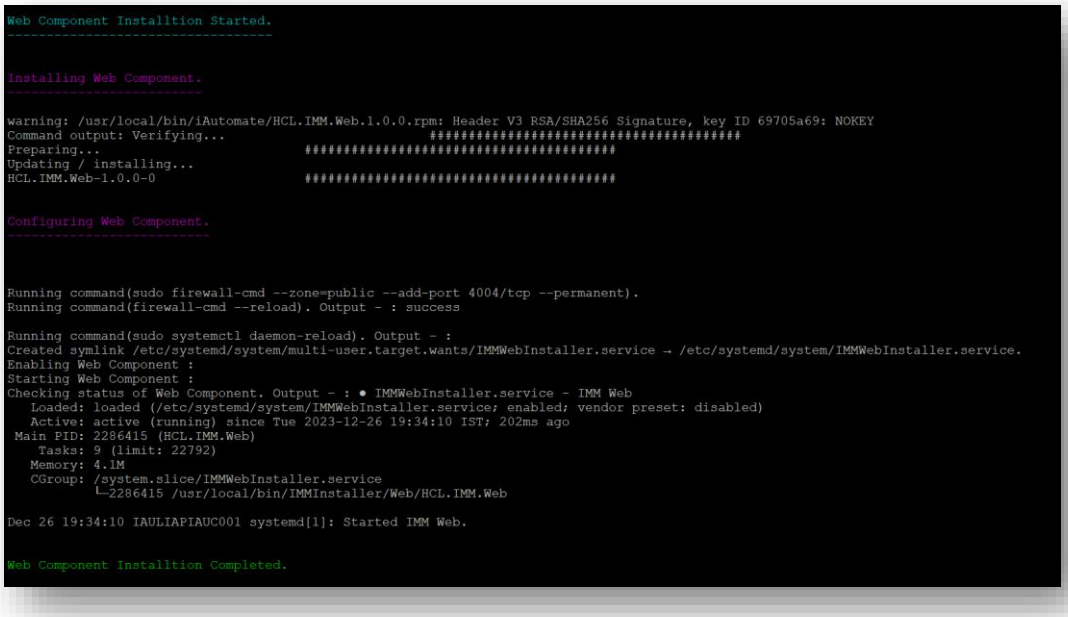


Figure 46 – Web Component installation.

19. The Installation success message appears. It also prints the website URL.

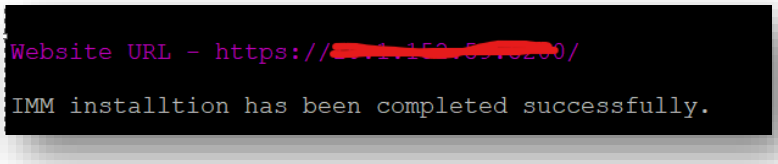


Figure 47 - Installation Success Message

20. Run Website URL and login with root user created while installing the IMM application.

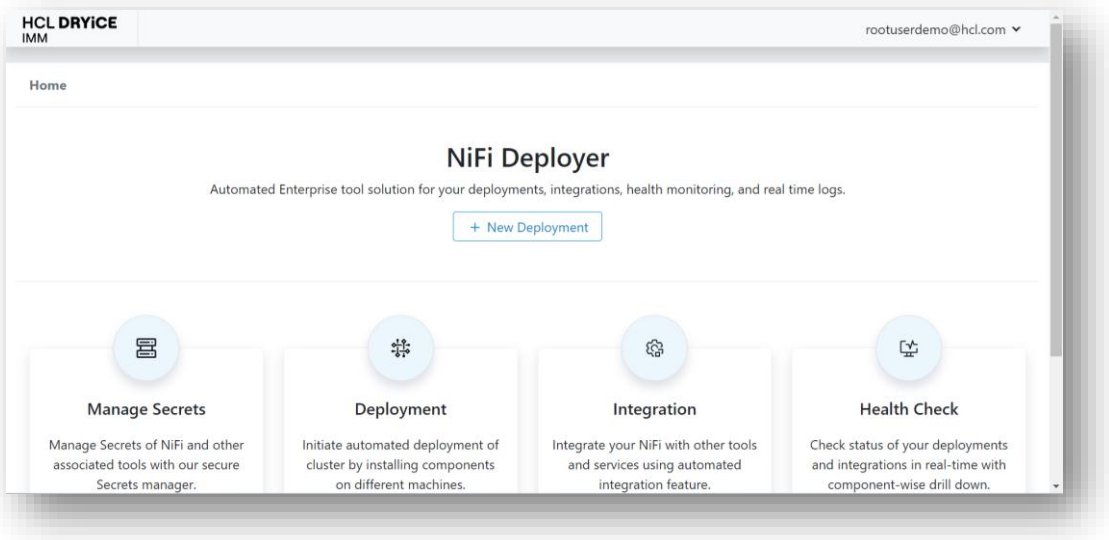


Figure 48 – IMM Application.

2.8.3 IMM Upgradation

This section lists the steps to upgrade IMM components on all Linux machines.

To upgrade IMM, perform the following steps:

1. When the installer is executed, it examines whether a prior version is present on the machine; if so, it initiates an upgrade installation. The following page appears:

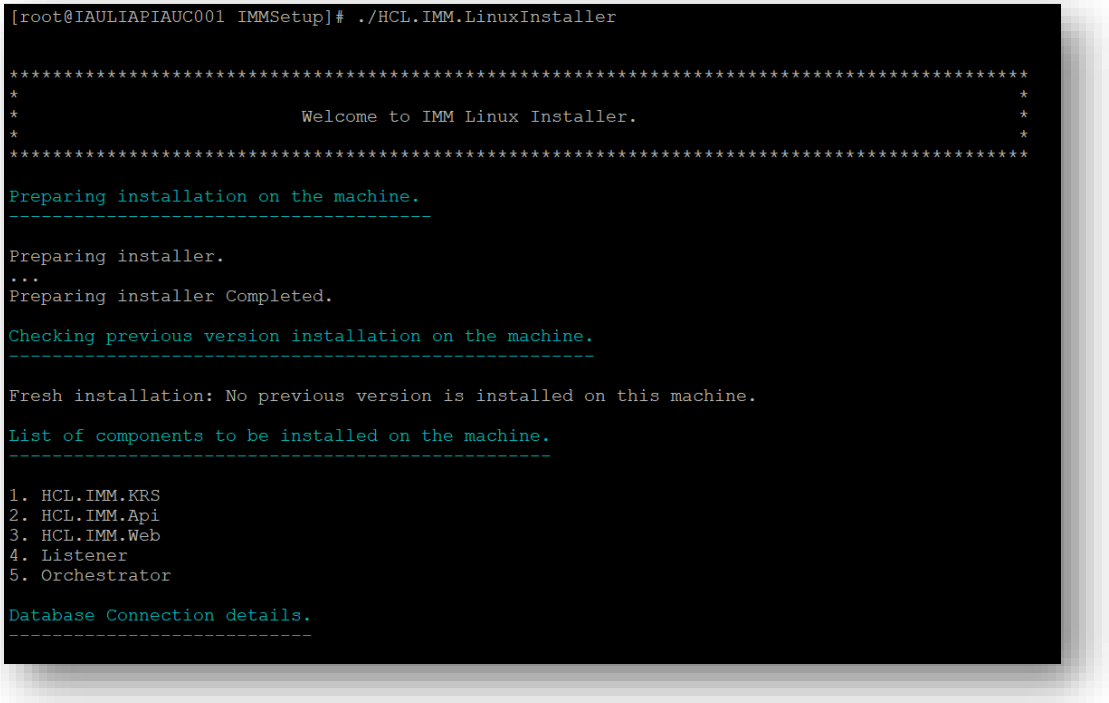


Figure 49 - IMM Installer – Upgrade

2. It seeks confirmation from the user regarding the components they want to install on the machine and during the upgrade process it exclusively updates the selected components. Refer Figure 49 - IMM Installer – Upgrade.
3. To access database, the installer requires **Database Connection Details**. Input your database server details.

   For validation: Put any value in this field as it has empty validation.

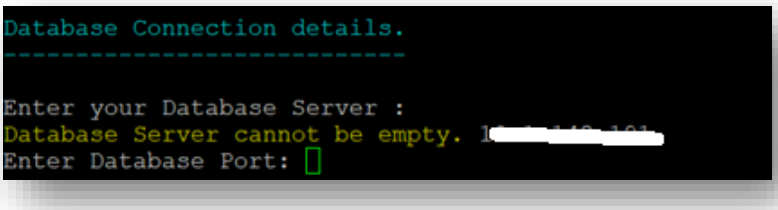


Figure 50 – Database Connection Details, Database Server

4. Enter the **Database Port**. If no port is provided, it uses the default port i.e.,5432.

   Validation: Put any valid number in this field as it has number only validation.

Port number entered should be greater than zero.

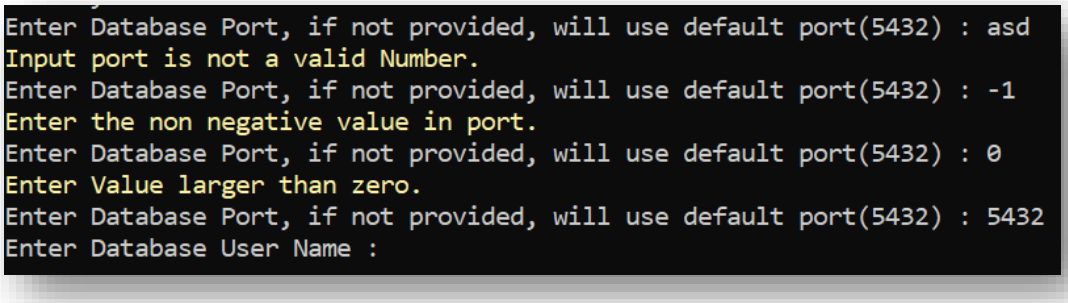


Figure 51 – Database Connection Details, Database Port.

5. Input **Database Username**.

   Validation: Put any value in this field as it has empty validation.

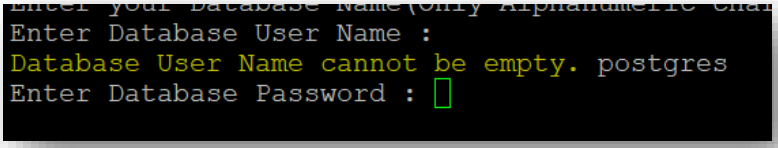


Figure 52 – Database Connection Details, Database Username.

6. Input **Database Password**.

   Validation: Put any value in this field as it has empty validation.

The input password is masked on screen for security purposes.

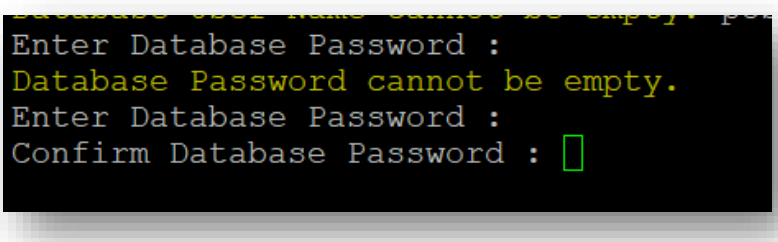


Figure 53 – Database Connection Details, Database Password

7. The installer requests a final confirmation from the user before proceeding with the installation.

```
Do you want to proceed with Installation? [y/n] y
```

Figure 54 - Installation confirmation before proceeding further

8. On selecting **Yes** by the user, the upgradation starts.

```
KRS Component Updation Started.
-------------------------------


Updating KRS Service.
---------------------

warning: /usr/local/bin/iAutomate/HCL.IMM.KRS.1.0.0.rpm: Header V3 RSA/SHA256 Signature, key ID 3a7e09e6: NOKEY
        package HCL.IMM.KRS-1.0.0-0.noarch is already installed
Command output: Verifying...                ########################################
Preparing...                          ########################################


Configuring KRS Service.
------------------------



Running command(firewall-cmd --reload). Output - : success

Running command(sudo systemctl daemon-reload). Output - :
Enabling KRS Service :
Starting KRS Service :
Checking status of KRS Service. Output - : ● IMMKRSInstaller.service - IMM KRS
   Loaded: loaded (/etc/systemd/system/IMMKRSInstaller.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2023-12-26 19:33:52 IST; 34min ago
 Main PID: 2285968 (HCL.IMM.KRS)
    Tasks: 18 (limit: 22792)
   Memory: 100.1M
   CGroup: /system.slice/IMMKRSInstaller.service
           └─2285968 /usr/local/bin/IMMInstaller/KRS/HCL.IMM.KRS

Dec 26 19:33:52 IAULIAPIAUC001 systemd[1]: Started IMM KRS.
Dec 26 19:34:03 IAULIAPIAUC001 IMMKRS[2285968]: 2023-12-26 19:34:03 [INF] Start AddApplicationKey
Dec 26 19:34:03 IAULIAPIAUC001 IMMKRS[2285968]: 2023-12-26 19:34:03 [INF] END AddApplicationKey
Dec 26 19:34:03 IAULIAPIAUC001 IMMKRS[2285968]: 2023-12-26 19:34:03 [INF] Start AddApplicationKey
Dec 26 19:34:03 IAULIAPIAUC001 IMMKRS[2285968]: 2023-12-26 19:34:03 [INF] END AddApplicationKey


KRS Component Update Completed.
```

Figure 55 – KRS Component Upgradation

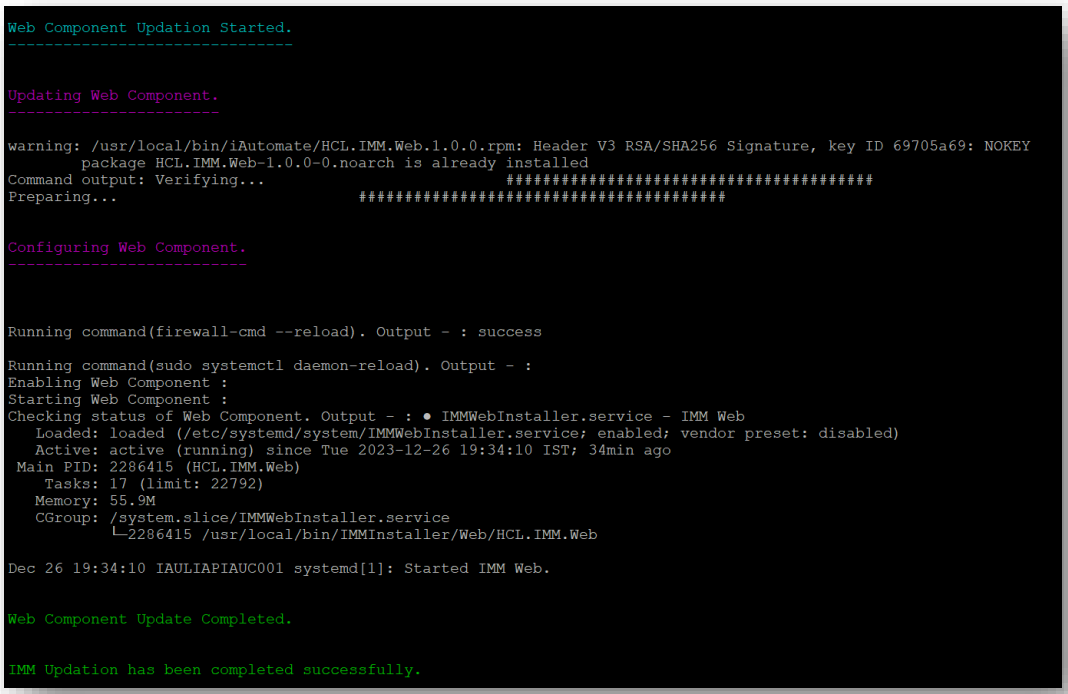


Figure 56 – Web Component Upgradation

2.8.4 IMM Uninstallation

This section lists the steps to uninstall IMM components on all Linux machines.

To uninstall IMM, perform the following steps:

1. login to the Linux server and go to the Installer folder. Refer to the section Run the Installer (*Step 1 to 3*).
2. Copy the UninstallIMM.sh file to installer folder.
3. Run the following command to uninstall IMM from the machine.

```
bash UninstallIMM.sh
```

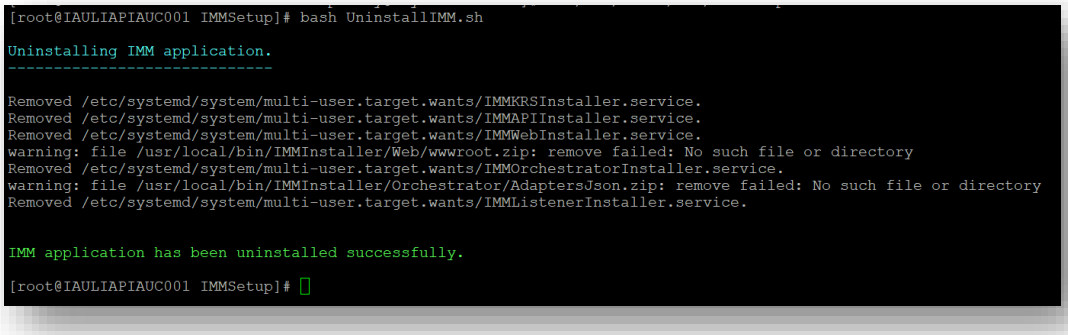


Figure 57 – IMM Components Uninstallation.

4. The IMM application is uninstalled successfully.

HCLSoftware

hcltechsw.com