HCLSoftware

HCL IntelliOps
Event Management

Collector Installation Guide

Version 1.2

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.

Version 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

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.

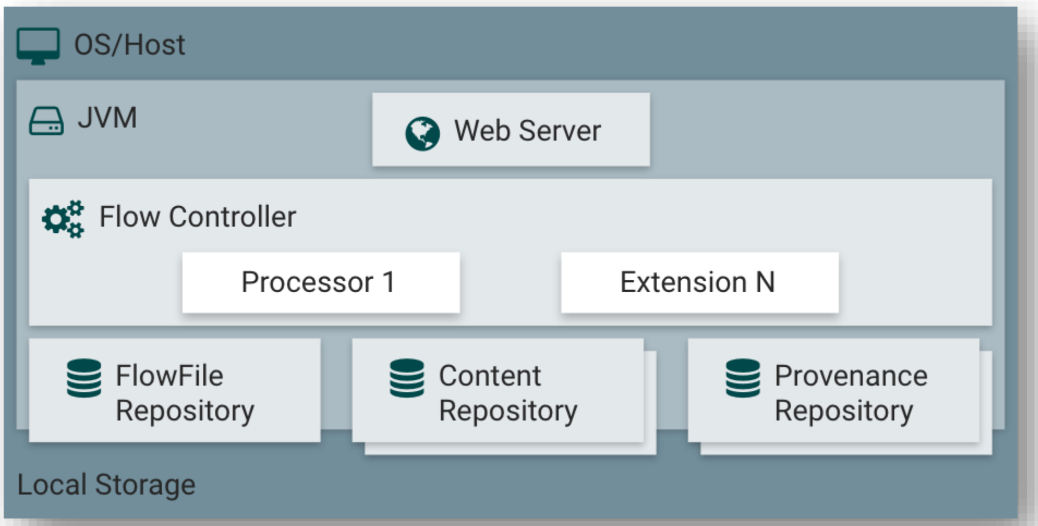


Figure 1 - Content and Provenance repository

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.

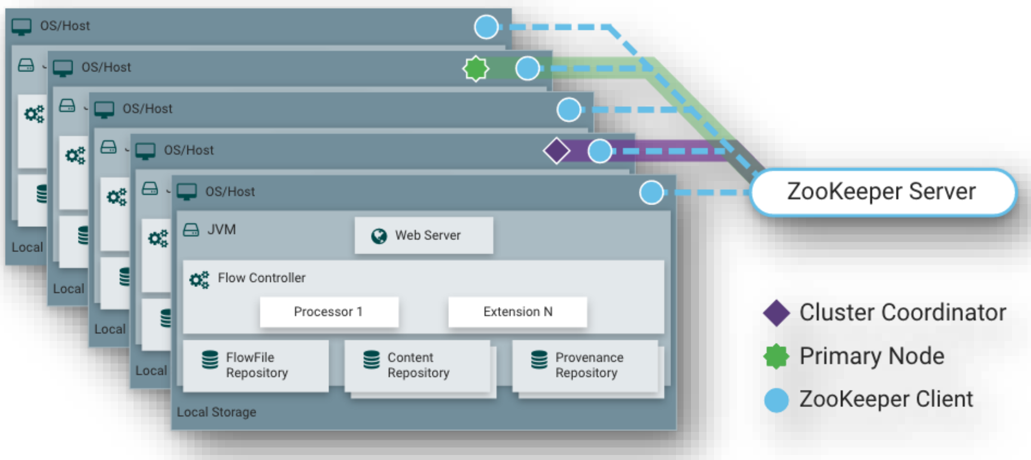


Figure 2 - Repositories

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 OS for NIFI

Linux RHEL 8 (Recommended)

- Unix
- Windows
- macOS

Requires Java 8 or Java 11

2.2.3 Supported Web Browsers

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

2.2.4 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

2.2.5 Port Requirement for NiFi

The following ports are required for internal communication:

- Nifi remote socket port = 10443
- Nifi web https port = 9443
- Nifi cluster port = 11443
- Nifi cluster load balance port = 6342
- Nifi zookeeper connect port=2181, 2888, 3888

2.4 NiFi Installation and Setup

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

2.4.1 Pre-requisites of NiFi Installation

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

1. Create user on all mentioned servers and to be named as "nifiadmin". Common credentials used for creation are (nifiadmin/XXXXX)

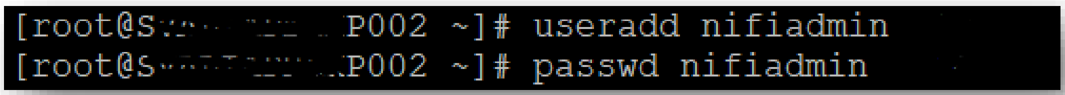
2. Run below command (on all the nodes) to create a new user:

```
useradd nifiadmin
```

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

```
passwd nifiadmin
```

Sample Console:



```
[root@S[redacted]P002 ~]# useradd nifiadmin
[root@S[redacted]P002 ~]# passwd nifiadmin
```

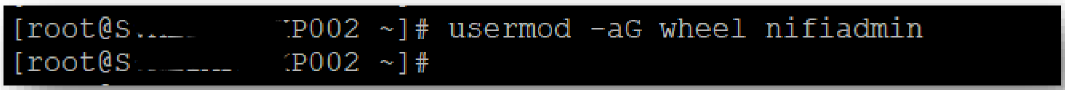
Perform all operations as NiFi admin user only.

4. The command below is used for modifying a user account, the -a switch tells the command to append, and the -G switch is telling you are using the group name.

5. Run below command (on all the nodes) for modifying a user account:

```
usermod -aG wheel nifiadmin
```

6. The wheel command is used to control access to the su or sudo command, which allows a user to masquerade as another user.

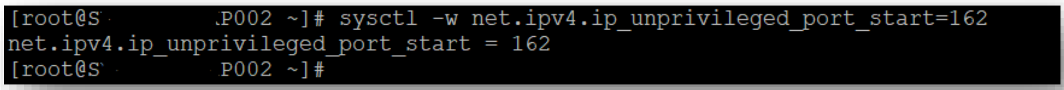


```
[root@S[redacted]P002 ~]# usermod -aG wheel nifiadmin
[root@S[redacted]P002 ~]#
```

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

7. Run below command (on all the nodes) for snmptrap port permission:

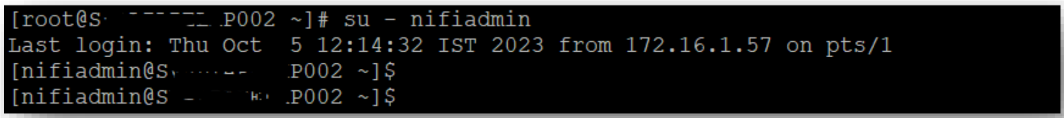
```
sysctl -w net. ipv4.ip_unprivileged_port_start=162
```



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

8. Switch to "nifiadmin" user to compete the installation steps.
9. Run the below command (on all nodes) for modifying a user account:

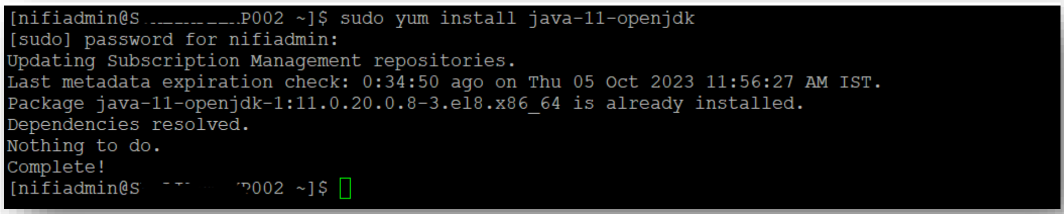
```
su – nifiadmin
```



```
[root@S[redacted]P002 ~]# su - nifiadmin
Last login: Thu Oct  5 12:14:32 IST 2023 from 172.16.1.57 on pts/1
[nifiadmin@S[redacted]P002 ~]$
[nifiadmin@S[redacted]P002 ~]$
```

10. Check if Java is installed on servers. If not, run the command below (on all the nodes) for JDK installation:

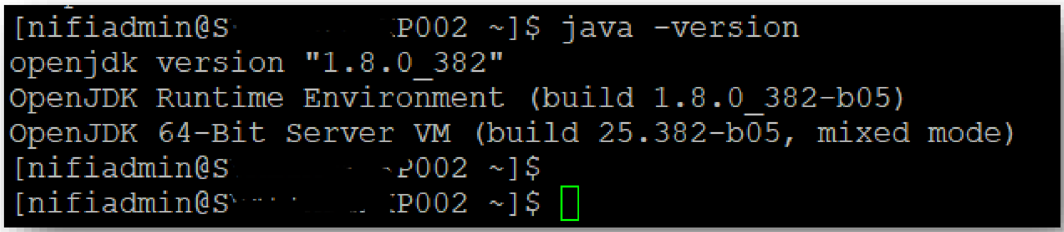
```
sudo yum install java-11-openjdk
```



```
[nifiadmin@S[redacted]P002 ~]$ sudo yum install java-11-openjdk
[sudo] password for nifiadmin:
Updating Subscription Management repositories.
Last metadata expiration check: 0:34:50 ago on Thu 05 Oct 2023 11:56:27 AM IST.
Package java-11-openjdk-1:11.0.20.0.8-3.el8.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[nifiadmin@S[redacted]P002 ~]$
```

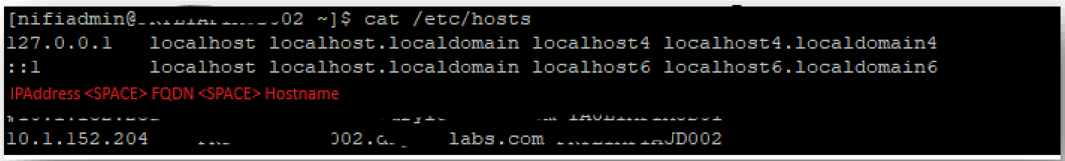
11. Validate the version of the java. Run the below command (on all nodes) check installed java version:

```
java -version
```



```
[nifiadmin@S[redacted]P002 ~]$ java -version
openjdk version "1.8.0_382"
OpenJDK Runtime Environment (build 1.8.0_382-b05)
OpenJDK 64-Bit Server VM (build 25.382-b05, mixed mode)
[nifiadmin@S[redacted]P002 ~]$
[nifiadmin@S[redacted]P002 ~]$
```

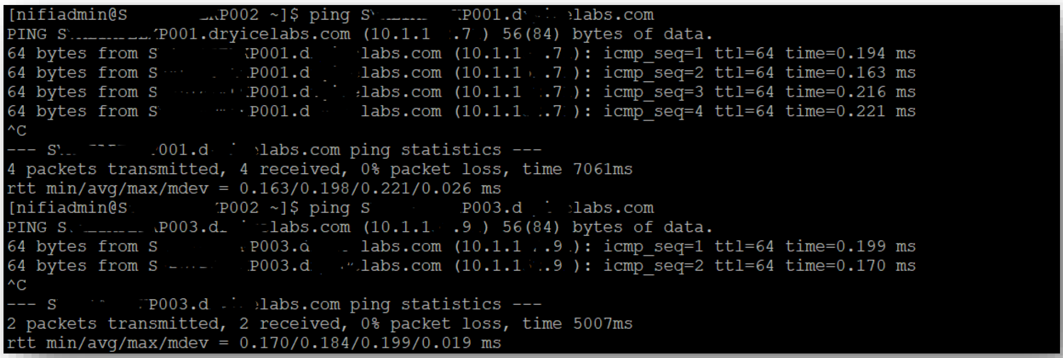
12. Ensure that the entry of /etc./hosts file is done on all the nodes as shown in the following screenshot:



```
[nifiadmin@[redacted]02 ~]$ cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
IPAddress <SPACE> FQDN <SPACE> Hostname
[redacted]
10.1.152.204    [redacted]002.d[redacted]labs.com [redacted]JD002
```

13. Do a ping check across all the nodes to confirm if the connectivity is established.
14. Run the below command (on all the nodes) to Ping check between all servers in cluster:

```
ping <server FQDN>
```



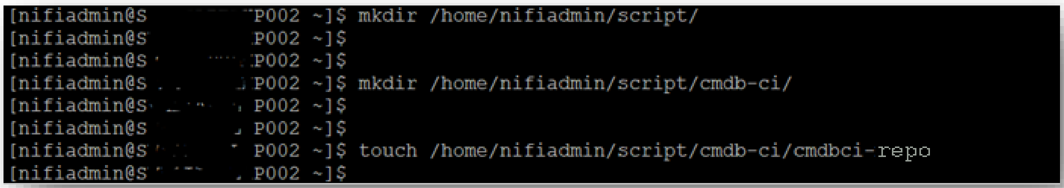
```
[nifiadmin@S[redacted]P002 ~]$ ping S[redacted]P001.d[redacted]labs.com
PING S[redacted]P001.dryicelabs.com (10.1.1[redacted].7 ) 56(84) bytes of data.
64 bytes from S[redacted]P001.d[redacted]labs.com (10.1.1[redacted].7 ): icmp_seq=1 ttl=64 time=0.194 ms
64 bytes from S[redacted]P001.d[redacted]labs.com (10.1.1[redacted].7 ): icmp_seq=2 ttl=64 time=0.163 ms
64 bytes from S[redacted]P001.d[redacted]labs.com (10.1.1[redacted].7 ): icmp_seq=3 ttl=64 time=0.216 ms
64 bytes from S[redacted]P001.d[redacted]labs.com (10.1.1[redacted].7 ): icmp_seq=4 ttl=64 time=0.221 ms
^C
--- S[redacted]001.d[redacted]labs.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 7061ms
rtt min/avg/max/mdev = 0.163/0.198/0.221/0.026 ms
[nifiadmin@S[redacted]P002 ~]$ ping S[redacted]P003.d[redacted]labs.com
PING S[redacted]P003.d[redacted]labs.com (10.1.1[redacted].9 ) 56(84) bytes of data.
64 bytes from S[redacted]P003.d[redacted]labs.com (10.1.1[redacted].9 ): icmp_seq=1 ttl=64 time=0.199 ms
64 bytes from S[redacted]P003.d[redacted]labs.com (10.1.1[redacted].9 ): icmp_seq=2 ttl=64 time=0.170 ms
^C
--- S[redacted]P003.d[redacted]labs.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 5007ms
rtt min/avg/max/mdev = 0.170/0.184/0.199/0.019 ms
```

15. Ensure that the following ports are open:
    - 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
16. Create directories and files required for integration:
17. Run the following commands (on All node) to create a directory "script" and then "cmdb-ci":

```
mkdir /home/nifiadmin/script/
mkdir /home/nifiadmin/script/cmdb-ci/
```

18. Create a file to store CMDB data:

```
touch/home/nifiadmin/script/cmdb-ci/cmdbci-repo
```



```
[nifiadmin@S[redacted]P002 ~]$ mkdir /home/nifiadmin/script/
[nifiadmin@S[redacted]P002 ~]$
[nifiadmin@S[redacted]P002 ~]$
[nifiadmin@S[redacted]P002 ~]$ mkdir /home/nifiadmin/script/cmdb-ci/
[nifiadmin@S[redacted]P002 ~]$
[nifiadmin@S[redacted]P002 ~]$
[nifiadmin@S[redacted]P002 ~]$ touch /home/nifiadmin/script/cmdb-ci/cmdbci-repo
[nifiadmin@S[redacted]P002 ~]$
```

2.4.2 Installation of NiFi Cluster

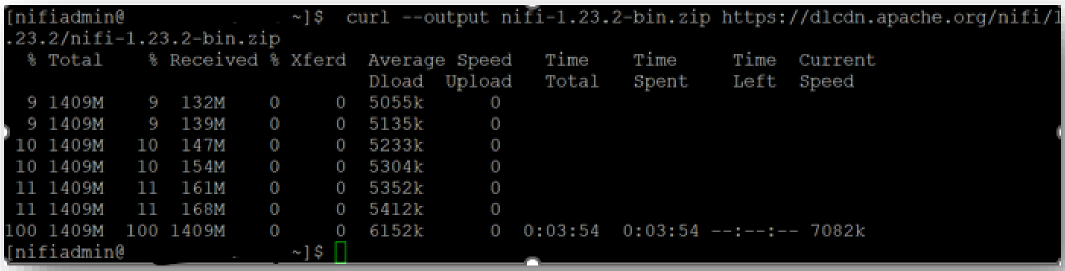
Below binaries are used during the installation of clusters.

– **Nifi Toolkit:** https://archive.apache.org/dist/nifi/1.23.2/nifi-toolkit-1.23.2-bin.zip

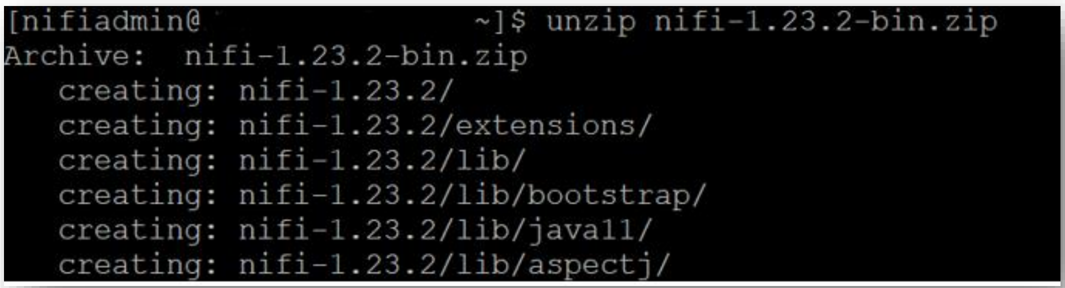
– **Nifi Binary:** https://archive.apache.org/dist/nifi/1.23.2/nifi-1.23.2-bin.zip

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

```
curl -output nifi-1.23.2-bin.zip
https://archive.apache.org/dist/nifi/1.23.2/nifi-1.23.2-bin.zip

unzip nifi-1.23.2-bin.zip
```



```
[nifiadmin@[redacted] ~]$  curl --output nifi-1.23.2-bin.zip https://dlcdn.apache.org/nifi/1
.23.2/nifi-1.23.2-bin.zip
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  9 1409M    9  132M    0     0  5055k      0
  9 1409M    9  139M    0     0  5135k      0
 10 1409M   10  147M    0     0  5233k      0
 10 1409M   10  154M    0     0  5304k      0
 11 1409M   11  161M    0     0  5352k      0
 11 1409M   11  168M    0     0  5412k      0
100 1409M  100 1409M    0     0  6152k      0  0:03:54  0:03:54 --:--:-- 7082k
[nifiadmin@[redacted] ~]$
```

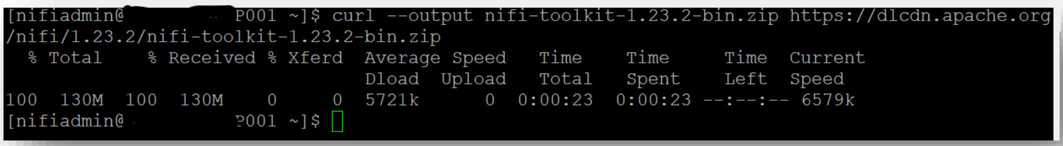


```
[nifiadmin@[redacted] ~]$ unzip nifi-1.23.2-bin.zip
Archive:  nifi-1.23.2-bin.zip
   creating: nifi-1.23.2/
   creating: nifi-1.23.2/extensions/
   creating: nifi-1.23.2/lib/
   creating: nifi-1.23.2/lib/bootstrap/
   creating: nifi-1.23.2/lib/java11/
   creating: nifi-1.23.2/lib/aspectj/
```

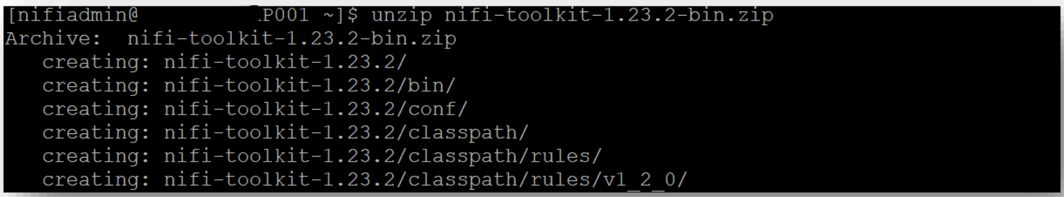
3. Download Nifi-toolkit only on primary server. Use this toolkit link and extract the package on all the nodes.

   To download Nifi-toolkit, run the following commands:

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

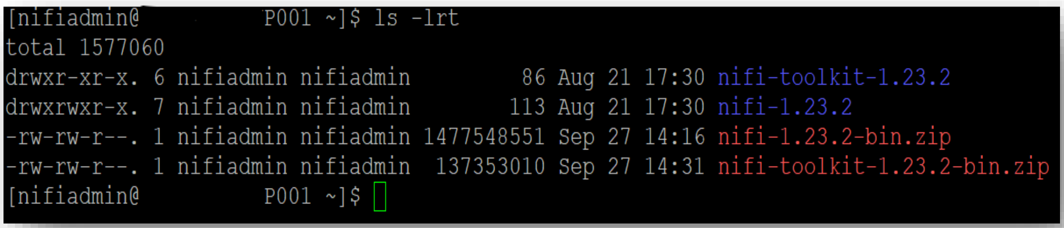


```
[nifiadmin@[redacted]P001 ~]$ curl --output nifi-toolkit-1.23.2-bin.zip https://dlcdn.apache.org
/nifi/1.23.2/nifi-toolkit-1.23.2-bin.zip
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  130M  100  130M    0     0  5721k      0  0:00:23  0:00:23 --:--:-- 6579k
[nifiadmin@[redacted]P001 ~]$
```



```
[nifiadmin@[redacted]P001 ~]$ unzip nifi-toolkit-1.23.2-bin.zip
Archive:  nifi-toolkit-1.23.2-bin.zip
   creating: nifi-toolkit-1.23.2/
   creating: nifi-toolkit-1.23.2/bin/
   creating: nifi-toolkit-1.23.2/conf/
   creating: nifi-toolkit-1.23.2/classpath/
   creating: nifi-toolkit-1.23.2/classpath/rules/
   creating: nifi-toolkit-1.23.2/classpath/rules/v1_2_0/
```

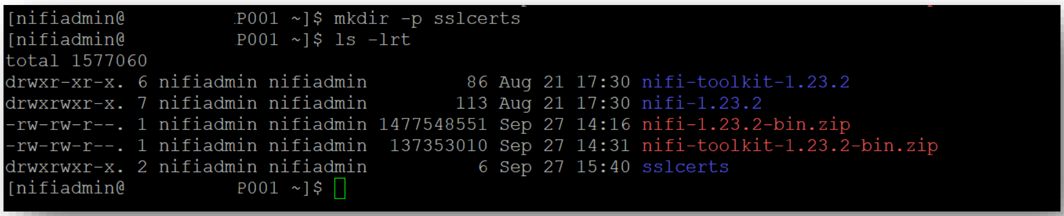
4. After extracting the folders, the files are listed as shown in the below figure:



```
[nifiadmin@[redacted]P001 ~]$ ls -lrt
total 1577060
drwxr-xr-x. 6 nifiadmin nifiadmin         86 Aug 21 17:30 nifi-toolkit-1.23.2
drwxrwxr-x. 7 nifiadmin nifiadmin        113 Aug 21 17:30 nifi-1.23.2
-rw-rw-r--. 1 nifiadmin nifiadmin 1477548551 Sep 27 14:16 nifi-1.23.2-bin.zip
-rw-rw-r--. 1 nifiadmin nifiadmin  137353010 Sep 27 14:31 nifi-toolkit-1.23.2-bin.zip
[nifiadmin@[redacted]P001 ~]$
```

5. Create self-signed certificates on the Primary Server for all the cluster servers. Run the following command on the Primary node:

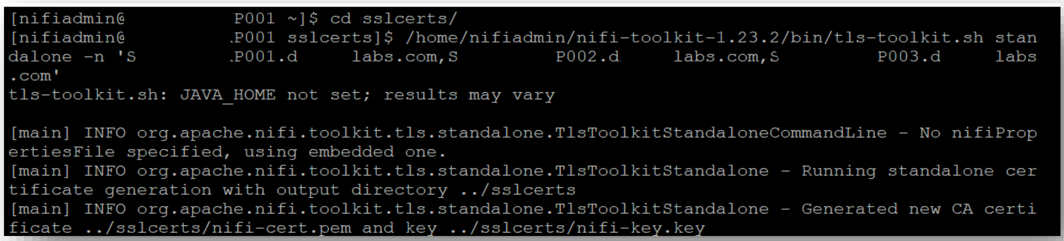
```
mkdir -p sslcerts
```



```
[nifiadmin@[redacted]P001 ~]$ mkdir -p sslcerts
[nifiadmin@[redacted]P001 ~]$ ls -lrt
total 1577060
drwxr-xr-x. 6 nifiadmin nifiadmin         86 Aug 21 17:30 nifi-toolkit-1.23.2
drwxrwxr-x. 7 nifiadmin nifiadmin        113 Aug 21 17:30 nifi-1.23.2
-rw-rw-r--. 1 nifiadmin nifiadmin 1477548551 Sep 27 14:16 nifi-1.23.2-bin.zip
-rw-rw-r--. 1 nifiadmin nifiadmin  137353010 Sep 27 14:31 nifi-toolkit-1.23.2-bin.zip
drwxrwxr-x. 2 nifiadmin nifiadmin          6 Sep 27 15:40 sslcerts
[nifiadmin@[redacted]P001 ~]$
```

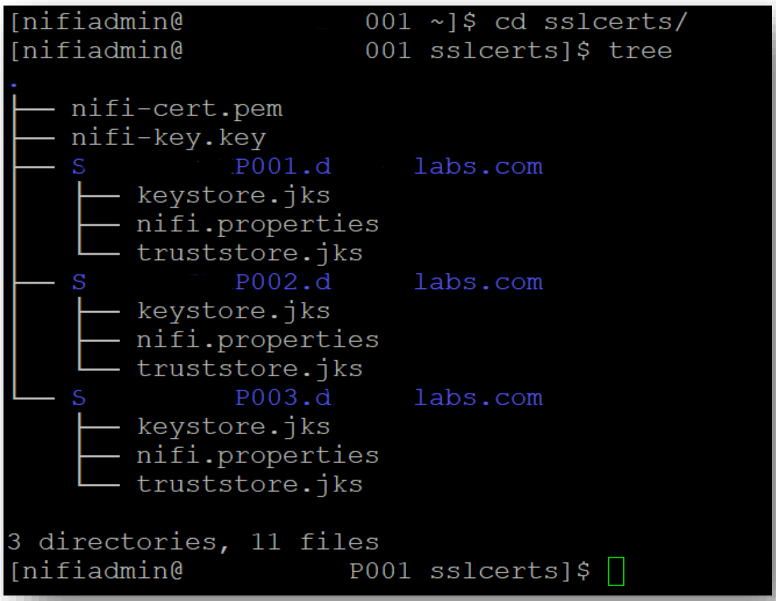
6. For generating the SSL certificate, run the following command on the Primary node:

```
cd sslcerts/

/home/nifiadmin/nifi-toolkit-1.23.2/bin/tls-toolkit.sh
standalone -n
'Server001.domain.com,Server002.domain.com,Server003.domain.com
'
```



```
[nifiadmin@          P001 ~]$ cd sslcerts/
[nifiadmin@         .P001 sslcerts]$ /home/nifiadmin/nifi-toolkit-1.23.2/bin/tls-toolkit.sh stan
dalone -n 'S         .P001.d     labs.com,S          P002.d     labs.com,S          P003.d     labs
.com'
tls-toolkit.sh: JAVA_HOME not set; results may vary

[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandaloneCommandLine - No nifiProp
ertiesFile specified, using embedded one.
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Running standalone cer
tificate generation with output directory ../sslcerts
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Generated new CA certi
ficate ../sslcerts/nifi-cert.pem and key ../sslcerts/nifi-key.key
```

7. The FQDNs (Fully Qualified Domain Name) are created with following structure along with. pem and .key files.



```
[nifiadmin@           001 ~]$ cd sslcerts/
[nifiadmin@           001 sslcerts]$ tree
.
├── nifi-cert.pem
├── nifi-key.key
├── S          P001.d     labs.com
│   ├── keystore.jks
│   ├── nifi.properties
│   └── truststore.jks
├── S          P002.d     labs.com
│   ├── keystore.jks
│   ├── nifi.properties
│   └── truststore.jks
└── S          P003.d     labs.com
    ├── keystore.jks
    ├── nifi.properties
    └── truststore.jks

3 directories, 11 files
[nifiadmin@          P001 sslcerts]$
```

8. Check the directory structure by using the following command:

```
[nifiadmin@Server001 sslcerts] $ tree
```

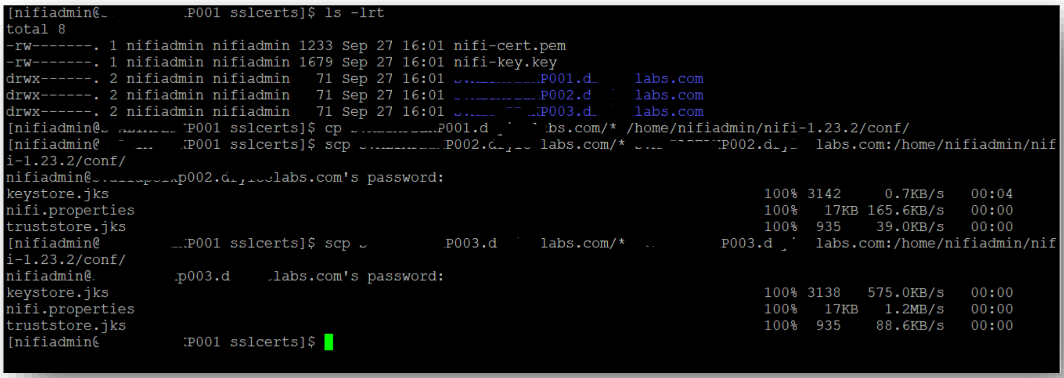
9. Copy the appropriate SSL folders to respective servers. Run the following commands on the Primary Node:
   - For copying to primary server:

```
cp Server001.domain.com/* /home/nifiadmin/nifi-1.23.2/conf/
```

   - For copying to other servers:

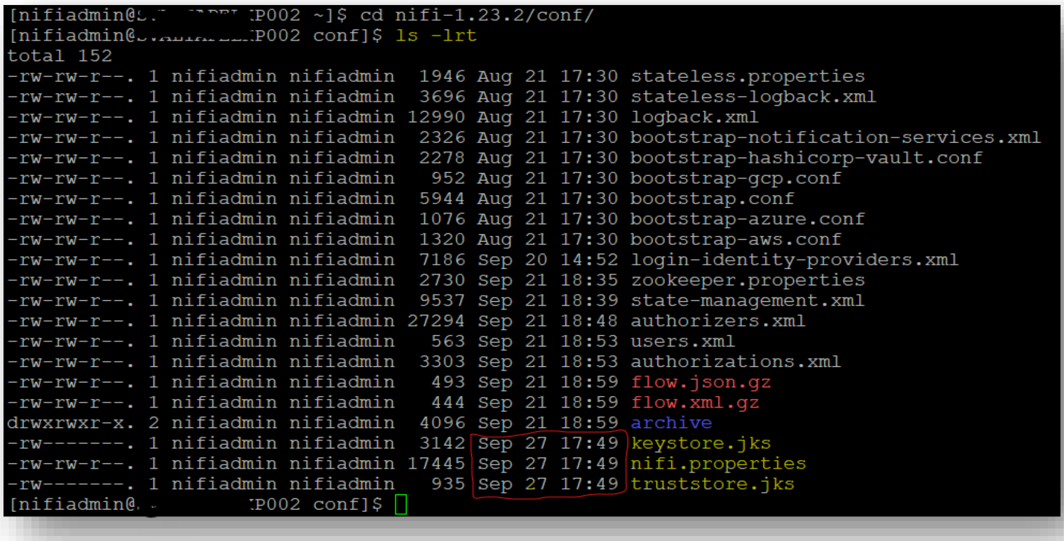
```
scp Server002.domain.com/*
Server002.domain.com:/home/nifiadmin/nifi-1.23.2/conf/
```

```
scp Server003.domain.com/*
Server003.domain.com:/home/nifiadmin/nifi-1.23.2/conf/
```



```
[nifiadmin@[redacted]P001 sslcerts]$ ls -lrt
total 8
-rw-------. 1 nifiadmin nifiadmin 1233 Sep 27 16:01 nifi-cert.pem
-rw-------. 1 nifiadmin nifiadmin 1679 Sep 27 16:01 nifi-key.key
drwx------. 2 nifiadmin nifiadmin   71 Sep 27 16:01 [redacted]P001.d[redacted]labs.com
drwx------. 2 nifiadmin nifiadmin   71 Sep 27 16:01 [redacted]P002.d[redacted]labs.com
drwx------. 2 nifiadmin nifiadmin   71 Sep 27 16:01 [redacted]P003.d[redacted]labs.com
[nifiadmin@[redacted]P001 sslcerts]$ cp [redacted]P001.d[redacted]bs.com/* /home/nifiadmin/nifi-1.23.2/conf/
[nifiadmin@[redacted]P001 sslcerts]$ scp [redacted]P002.d[redacted]labs.com/* [redacted]P002.d[redacted]labs.com:/home/nifiadmin/nif
i-1.23.2/conf/
nifiadmin@[redacted]p002.d[redacted]labs.com's password:
keystore.jks                                                                 100% 3142     0.7KB/s   00:04
nifi.properties                                                              100%   17KB 165.6KB/s   00:00
truststore.jks                                                               100%  935    39.0KB/s   00:00
[nifiadmin@[redacted]P001 sslcerts]$ scp [redacted]P003.d[redacted]labs.com/* [redacted]P003.d[redacted]labs.com:/home/nifiadmin/nif
i-1.23.2/conf/
nifiadmin@[redacted]p003.d[redacted]labs.com's password:
keystore.jks                                                                 100% 3138   575.0KB/s   00:00
nifi.properties                                                              100%   17KB   1.2MB/s   00:00
truststore.jks                                                               100%  935    88.6KB/s   00:00
[nifiadmin@[redacted]P001 sslcerts]$
```

10. If you get the above screenshot (100% transferred completed), that means file is transferred.
11. On successful transfer, the following screen appears and displays the updated files with time stamp.



```
[nifiadmin@[redacted]P002 ~]$ cd nifi-1.23.2/conf/
[nifiadmin@[redacted]P002 conf]$ ls -lrt
total 152
-rw-rw-r--. 1 nifiadmin nifiadmin  1946 Aug 21 17:30 stateless.properties
-rw-rw-r--. 1 nifiadmin nifiadmin  3696 Aug 21 17:30 stateless-logback.xml
-rw-rw-r--. 1 nifiadmin nifiadmin 12990 Aug 21 17:30 logback.xml
-rw-rw-r--. 1 nifiadmin nifiadmin  2326 Aug 21 17:30 bootstrap-notification-services.xml
-rw-rw-r--. 1 nifiadmin nifiadmin  2278 Aug 21 17:30 bootstrap-hashicorp-vault.conf
-rw-rw-r--. 1 nifiadmin nifiadmin   952 Aug 21 17:30 bootstrap-gcp.conf
-rw-rw-r--. 1 nifiadmin nifiadmin  5944 Aug 21 17:30 bootstrap.conf
-rw-rw-r--. 1 nifiadmin nifiadmin  1076 Aug 21 17:30 bootstrap-azure.conf
-rw-rw-r--. 1 nifiadmin nifiadmin  1320 Aug 21 17:30 bootstrap-aws.conf
-rw-rw-r--. 1 nifiadmin nifiadmin  7186 Sep 20 14:52 login-identity-providers.xml
-rw-rw-r--. 1 nifiadmin nifiadmin  2730 Sep 21 18:35 zookeeper.properties
-rw-rw-r--. 1 nifiadmin nifiadmin  9537 Sep 21 18:39 state-management.xml
-rw-rw-r--. 1 nifiadmin nifiadmin 27294 Sep 21 18:48 authorizers.xml
-rw-rw-r--. 1 nifiadmin nifiadmin   563 Sep 21 18:53 users.xml
-rw-rw-r--. 1 nifiadmin nifiadmin  3303 Sep 21 18:53 authorizations.xml
-rw-rw-r--. 1 nifiadmin nifiadmin   493 Sep 21 18:59 flow.json.gz
-rw-rw-r--. 1 nifiadmin nifiadmin   444 Sep 21 18:59 flow.xml.gz
drwxrwxr-x. 2 nifiadmin nifiadmin  4096 Sep 21 18:59 archive
-rw-------. 1 nifiadmin nifiadmin  3142 Sep 27 17:49 keystore.jks
-rw-rw-r--. 1 nifiadmin nifiadmin 17445 Sep 27 17:49 nifi.properties
-rw-------. 1 nifiadmin nifiadmin   935 Sep 27 17:49 truststore.jks
[nifiadmin@[redacted]P002 conf]$
```

12. Run the following command (on all the nodes) to configure the zookeeper properties.

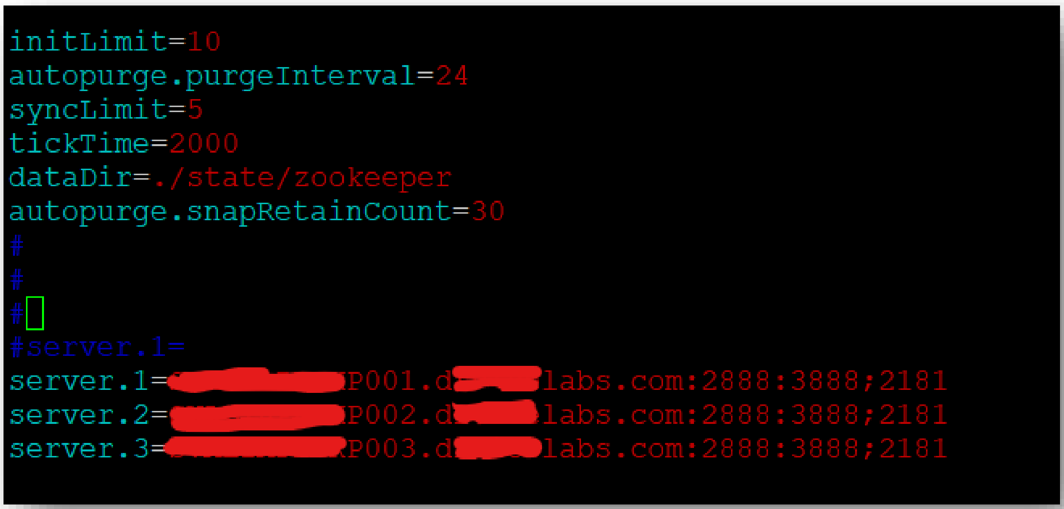
```
Change the directory: cd /home/nifiadmin/nifi-1.23.2/conf/
```

13. Edit in zookeeper.properties as per the following table (only the lines marked in bold need to be updated):

```
vim zookeeper.properties
```

Table 3 - Zookeeper.Properties

```
Path:  /home/nifiadmin/nifi-1.23.2/conf/ zookeeper.properties
initLimit=10
autopurge.purgeInterval=24
syncLimit=5
tickTime=2000
dataDir=. /state/zookeeper
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
```

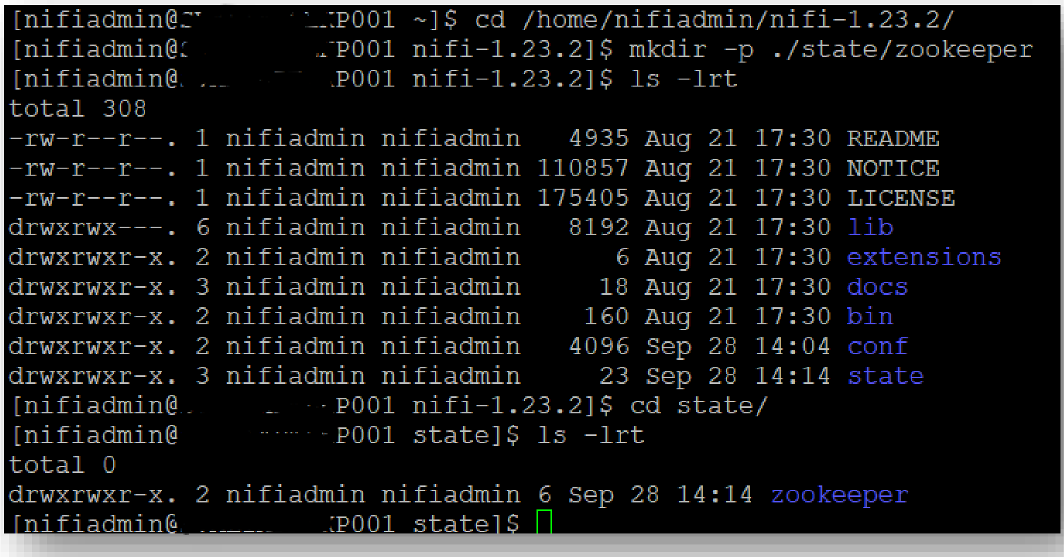


```
initLimit=10
autopurge.purgeInterval=24
syncLimit=5
tickTime=2000
dataDir=./state/zookeeper
autopurge.snapRetainCount=30
#
#
#
#server.1=
server.1=[redacted]P001.d[redacted]labs.com:2888:3888;2181
server.2=[redacted]P002.d[redacted]labs.com:2888:3888;2181
server.3=[redacted]P003.d[redacted]labs.com:2888:3888;2181
```

14. To create a **/state/zookeeper** directory, run the below commands (on all the nodes) on the path **nifi-1.23.2/**.

```
cd /home/nifiadmin/nifi-1.23.2/
```

```
mkdir -p./state/zookeeper
```



```
[nifiadmin@[redacted]P001 ~]$ cd /home/nifiadmin/nifi-1.23.2/
[nifiadmin@[redacted]P001 nifi-1.23.2]$ mkdir -p ./state/zookeeper
[nifiadmin@[redacted]P001 nifi-1.23.2]$ ls -lrt
total 308
-rw-r--r--. 1 nifiadmin nifiadmin   4935 Aug 21 17:30 README
-rw-r--r--. 1 nifiadmin nifiadmin 110857 Aug 21 17:30 NOTICE
-rw-r--r--. 1 nifiadmin nifiadmin 175405 Aug 21 17:30 LICENSE
drwxrwx---. 6 nifiadmin nifiadmin   8192 Aug 21 17:30 lib
drwxrwxr-x. 2 nifiadmin nifiadmin      6 Aug 21 17:30 extensions
drwxrwxr-x. 3 nifiadmin nifiadmin     18 Aug 21 17:30 docs
drwxrwxr-x. 2 nifiadmin nifiadmin    160 Aug 21 17:30 bin
drwxrwxr-x. 2 nifiadmin nifiadmin   4096 Sep 28 14:04 conf
drwxrwxr-x. 3 nifiadmin nifiadmin     23 Sep 28 14:14 state
[nifiadmin@[redacted]P001 nifi-1.23.2]$ cd state/
[nifiadmin@[redacted]P001 state]$ ls -lrt
total 0
drwxrwxr-x. 2 nifiadmin nifiadmin 6 Sep 28 14:14 zookeeper
[nifiadmin@[redacted]P001 state]$
```

15. 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.

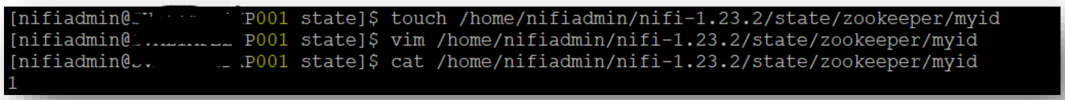
    For example: Set the value in myid file as 1 in SVXXX, 2 in SVXXX and 3 in SVXXX.

16. Run the following command to set the values on all the nodes:

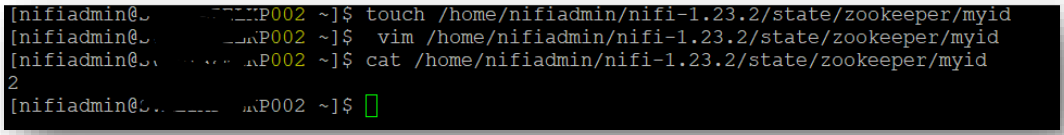
```
touch/home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
```

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

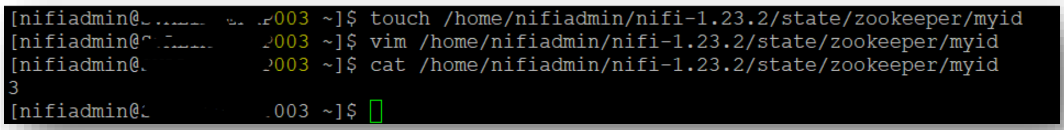
```
vim /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
```



```
[nifiadmin@[redacted]P001 state]$ touch /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@[redacted]P001 state]$ vim /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@[redacted]P001 state]$ cat /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
1
```



```
[nifiadmin@[redacted]P002 ~]$ touch /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@[redacted]P002 ~]$  vim /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@[redacted]P002 ~]$ cat /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
2
[nifiadmin@[redacted]P002 ~]$
```



```
[nifiadmin@[redacted]003 ~]$ touch /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@[redacted]003 ~]$ vim /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@[redacted]003 ~]$ cat /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
3
[nifiadmin@[redacted]003 ~]$
```

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

```
vim /home/nifiadmin/nifi-1.23.2/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</
class>
        <property name="Connect
String">Server001.domain.com:2181,
Server002.domain.com:2181,Server003.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>
```

19. Set the configuration below in nifi.properties, which were generated from ssl certificate on all the nodes. Only the lines marked in **bold** need to be updated.

```
vim /home/nifiadmin/nifi-1.23.2/conf/nifi.properties
```

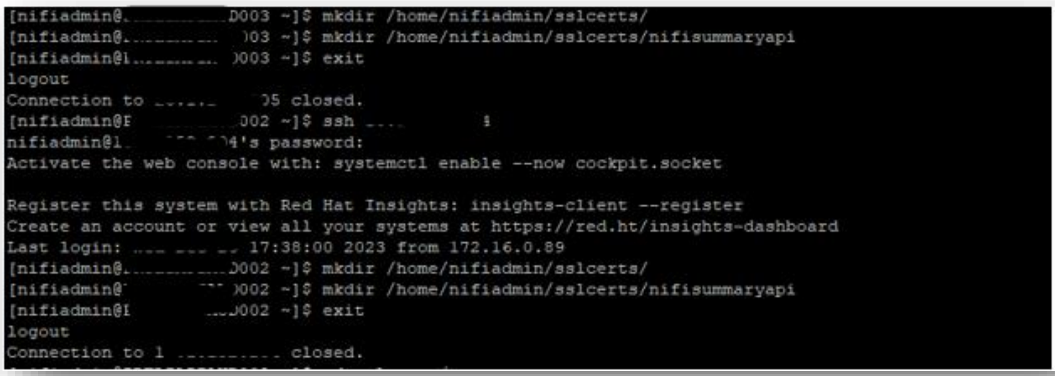
```
nifi.state.management.embedded.zookeeper.start=true
nifi.remote.input.host= Server001/002/003.domain.com
nifi.remote.input.secure=true
nifi.remote.input.socket.port=10443
nifi.remote.input.http.enabled=true
nifi.web.https.host=Server001/002/003.domain.com (Server
FQDN)
nifi.web.https.port=9443
nifi.web.
proxy.host=localhost:9443,Server001/002/003.domain.com:9443
(Server FQDN)
nifi.sensitive.props.key=propkeywith12chars
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= Server001/002/003.domain.com
(Server FQDN) nifi.cluster.load.balance.port=6342
nifi.zookeeper.connect.string=Svxxx.xyz.com:2181,
Svxxx.xyz.com:2181, Svxxx.xyz.com:2181
```

20. Create Nifi Self-signed certificate for SSL connection by using the following commands:

- **For nodes other than Primary**:

```
mkdir /home/nifiadmin/sslcerts/
```

```
mkdir /home/nifiadmin/sslcerts/nifisummaryapi/
```



```
[nifiadmin@[redacted]D003 ~]$ mkdir /home/nifiadmin/sslcerts/
[nifiadmin@[redacted])03 ~]$ mkdir /home/nifiadmin/sslcerts/nifisummaryapi
[nifiadmin@[redacted])003 ~]$ exit
logout
Connection to [redacted])5 closed.
[nifiadmin@[redacted]002 ~]$ ssh [redacted]
nifiadmin@[redacted]4's password:
Activate the web console with: systemctl enable --now cockpit.socket

Register this system with Red Hat Insights: insights-client --register
Create an account or view all your systems at https://red.ht/insights-dashboard
Last login: [redacted] 17:38:00 2023 from 172.16.0.89
[nifiadmin@[redacted]D002 ~]$ mkdir /home/nifiadmin/sslcerts/
[nifiadmin@[redacted])002 ~]$ mkdir /home/nifiadmin/sslcerts/nifisummaryapi
[nifiadmin@[redacted])002 ~]$ exit
logout
Connection to [redacted] closed.
```

- **For primary node**:

```
cd /home/nifiadmin/sslcerts/
```

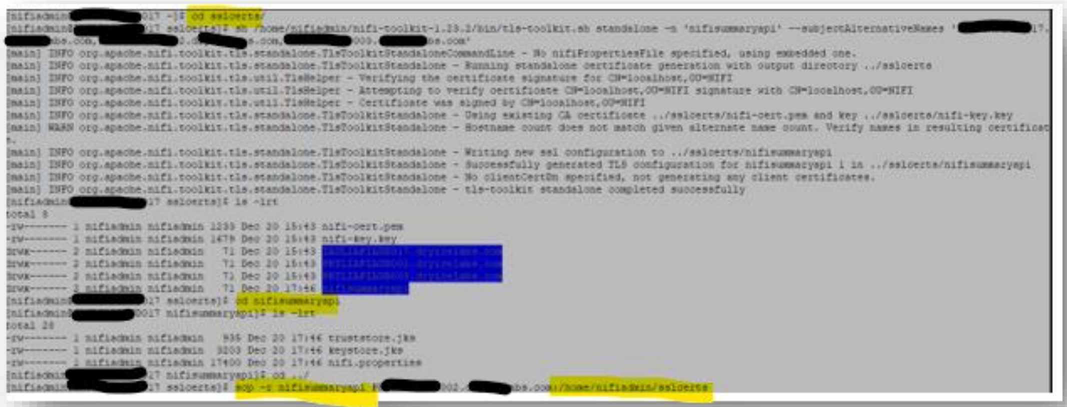
21. Run below commands to create self-signed certificate:

```
sh /home/nifiadmin/nifi-toolkit-1.23.2/bin/tls-toolkit.sh
standalone -n 'nifisummaryapi' --subjectAlternativeNames
'Server1.domain.com, Server2.domain.com, Server3.domain.com'
```

22. Copy the nifi-toolkit to Server002/ Server003:

```
scp -r nifisummaryapi
Server2.domain.com:/home/nifiadmin/sslcerts
```

```
scp -r nifisummaryapi
Server3.domain.com:/home/nifiadmin/sslcerts
```



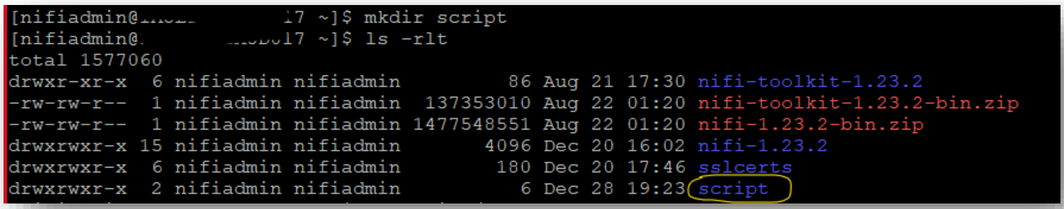
```
[nifiadmin[redacted]017 ~]$ cd sslcerts/
[nifiadmin@[redacted]17 sslcerts]$ sh /home/nifiadmin/nifi-toolkit-1.23.2/bin/tls-toolkit.sh standalone -n 'nifisummaryapi' --subjectAlternativeNames '[redacted]17.
[redacted]bs.com,[redacted]2.d[redacted]s.com,[redacted]003.[redacted]bs.com'
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandaloneCommandLine - No nifiPropertiesFile specified, using embedded one.
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Running standalone certificate generation with output directory ../sslcerts
[main] INFO org.apache.nifi.toolkit.tls.util.TlsHelper - Verifying the certificate signature for CN=localhost,OU=NIFI
[main] INFO org.apache.nifi.toolkit.tls.util.TlsHelper - Attempting to verify certificate CN=localhost,OU=NIFI signature with CN=localhost,OU=NIFI
[main] INFO org.apache.nifi.toolkit.tls.util.TlsHelper - Certificate was signed by CN=localhost,OU=NIFI
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Using existing CA certificate ../sslcerts/nifi-cert.pem and key ../sslcerts/nifi-key.key
[main] WARN org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Hostname count does not match given alternate name count. Verify names in resulting certificat
e.
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Writing new ssl configuration to ../sslcerts/nifisummaryapi
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Successfully generated TLS configuration for nifisummaryapi 1 in ../sslcerts/nifisummaryapi
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - No clientCertDn specified, not generating any client certificates.
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - tls-toolkit standalone completed successfully
[nifiadmin[redacted]17 sslcerts]$ ls -lrt
total 8
-rw------- 1 nifiadmin nifiadmin 1233 Dec 20 15:43 nifi-cert.pem
-rw------- 1 nifiadmin nifiadmin 1679 Dec 20 15:43 nifi-key.key
drwx------ 2 nifiadmin nifiadmin   71 Dec 20 15:43 [illegible]
drwx------ 2 nifiadmin nifiadmin   71 Dec 20 15:43 [illegible]
drwx------ 2 nifiadmin nifiadmin   71 Dec 20 15:43 [illegible]
drwx------ 2 nifiadmin nifiadmin   71 Dec 20 17:46 nifisummaryapi
[nifiadmin@[redacted]017 sslcerts]$ cd nifisummaryapi
[nifiadmin@[redacted]0017 nifisummaryapi]$ ls -lrt
total 28
-rw------- 1 nifiadmin nifiadmin   935 Dec 20 17:46 truststore.jks
-rw------- 1 nifiadmin nifiadmin  3203 Dec 20 17:46 keystore.jks
-rw------- 1 nifiadmin nifiadmin 17400 Dec 20 17:46 nifi.properties
[nifiadmin[redacted]17 nifisummaryapi]$ cd ../
[nifiadmin[redacted]17 sslcerts]$ scp -r nifisummaryapi [redacted]002.[redacted]bs.com:/home/nifiadmin/sslcerts
```

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

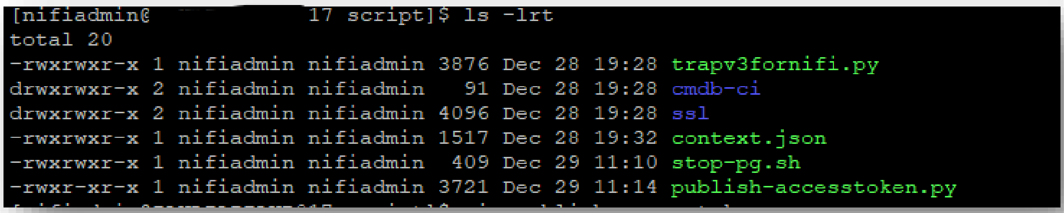
```
mkdir /home/nifiadmin/script/
```

24. 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

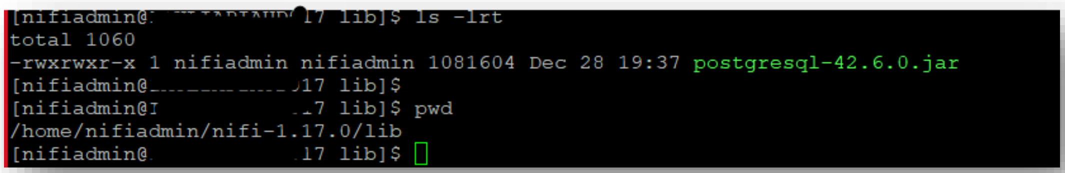


```
[nifiadmin@[redacted]17 ~]$ mkdir script
[nifiadmin@[redacted]17 ~]$ ls -rlt
total 1577060
drwxr-xr-x  6 nifiadmin nifiadmin         86 Aug 21 17:30 nifi-toolkit-1.23.2
-rw-rw-r--  1 nifiadmin nifiadmin  137353010 Aug 22 01:20 nifi-toolkit-1.23.2-bin.zip
-rw-rw-r--  1 nifiadmin nifiadmin 1477548551 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@[redacted]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
```

25. Copy Postgres jar file for DB connection on all nodes.

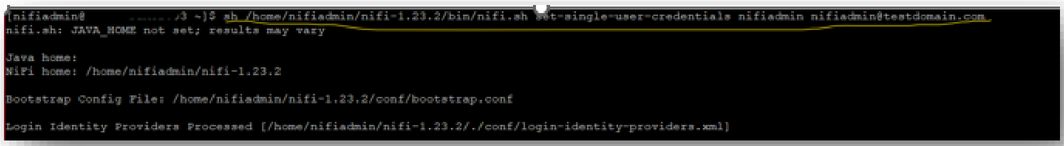


```
[nifiadmin@[redacted]17 lib]$ ls -lrt
total 1060
-rwxrwxr-x 1 nifiadmin nifiadmin 1081604 Dec 28 19:37 postgresql-42.6.0.jar
[nifiadmin@[redacted]17 lib]$
[nifiadmin@[redacted]7 lib]$ pwd
/home/nifiadmin/nifi-1.17.0/lib
[nifiadmin@[redacted]17 lib]$
```

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

Password length must be 14 characters:

```
sh /home/nifiadmin/nifi-1.23.2/bin/nifi.sh set-single-user-
credentials <username> <password@14letter>
```



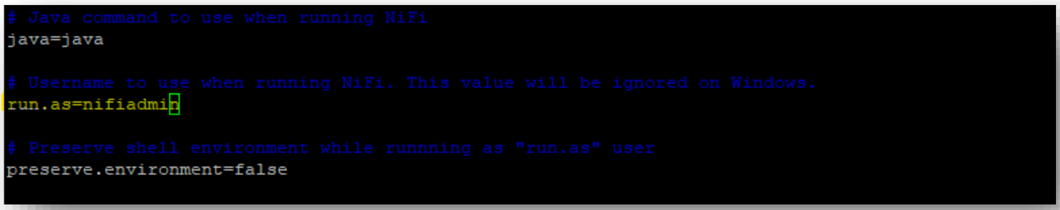
```
[nifiadmin@[redacted]3 ~]$ sh /home/nifiadmin/nifi-1.23.2/bin/nifi.sh set-single-user-credentials nifiadmin nifiadmin@testdomain.com
nifi.sh: JAVA_HOME not set; results may vary

Java home:
NiFi home: /home/nifiadmin/nifi-1.23.2

Bootstrap Config File: /home/nifiadmin/nifi-1.23.2/conf/bootstrap.conf

Login Identity Providers Processed [/home/nifiadmin/nifi-1.23.2/./conf/login-identity-providers.xml]
```

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

```
vim/home/nifiadmin/nifi-1.23.2/conf/bootstrap.conf
```

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

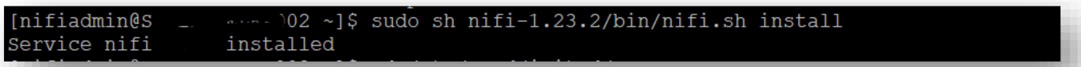
```
run.as: nifiadmin
```



```
# 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
```

29. Run the following on all the nodes to install Nifi as a service:

```
sudo sh /home/nifiadmin/nifi-1.23.2/bin/nifi.sh install
```



```
[nifiadmin@[redacted]02 ~]$ sudo sh nifi-1.23.2/bin/nifi.sh install
Service nifi [redacted] installed
```

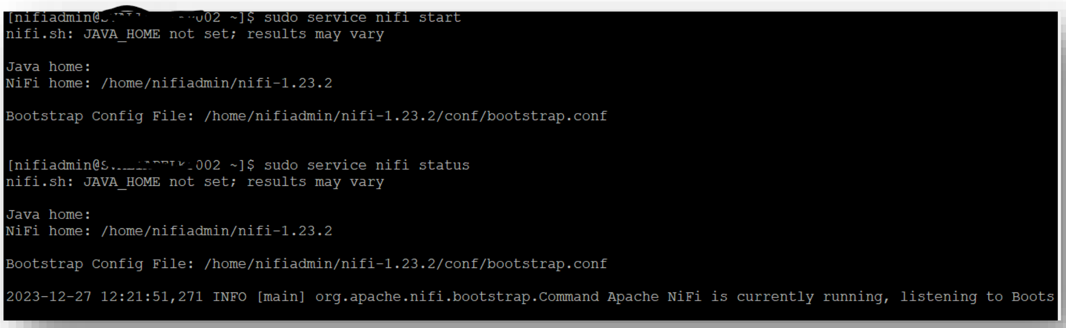
30. Run the following command on all the nodes to change the permission.

```
sudo chmod 755 /etc/rc.d/init.d/nifi
```

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

```
sudo service nifi start
```

```
sudo service nifi status
```



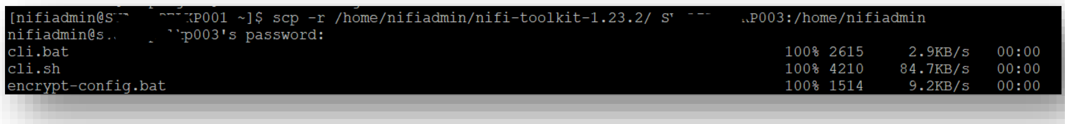
```
[nifiadmin@[redacted]002 ~]$ sudo service nifi start
nifi.sh: JAVA_HOME not set; results may vary

Java home:
NiFi home: /home/nifiadmin/nifi-1.23.2

Bootstrap Config File: /home/nifiadmin/nifi-1.23.2/conf/bootstrap.conf


[nifiadmin@S[redacted]002 ~]$ sudo service nifi status
nifi.sh: JAVA_HOME not set; results may vary

Java home:
NiFi home: /home/nifiadmin/nifi-1.23.2

Bootstrap Config File: /home/nifiadmin/nifi-1.23.2/conf/bootstrap.conf

2023-12-27 12:21:51,271 INFO [main] org.apache.nifi.bootstrap.Command Apache NiFi is currently running, listening to Boots
```

32. To encrypt all the passwords in Nifi configuration files, perform the below steps:
    a. Copy the encryption script from nifi-toolkit (which is downloaded on Primary server) to other nodes in Nifi cluster. Run the following commands only on primary node:

```
scp -r /home/nifiadmin/nifi-toolkit-1.23.2/
Server002:/home/nifiadmin
```

```
scp -r /home/nifiadmin/nifi-toolkit-1.23.2/
Server003:/home/nifiadmin
```

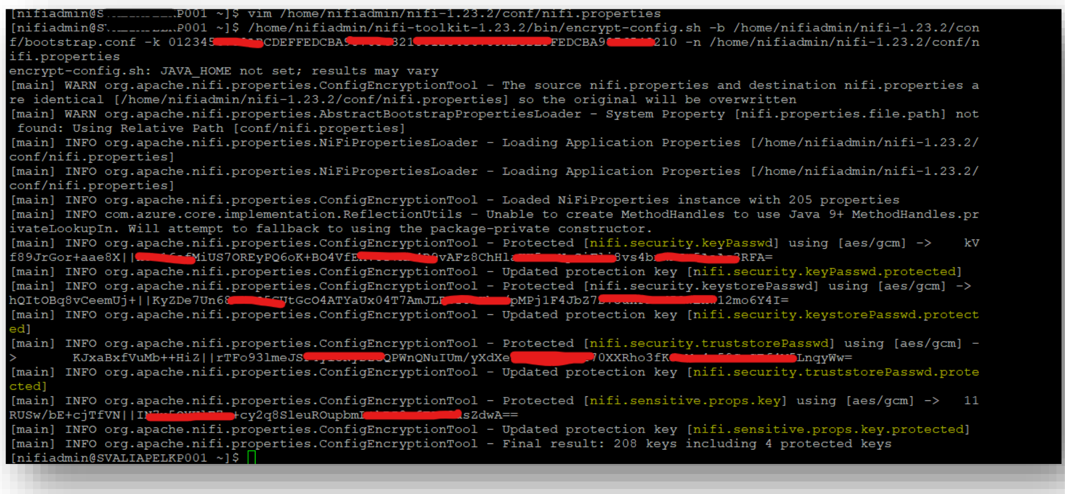
Copy the same nifi-toolkit on other servers in cluster just by changing server name.



```
[nifiadmin@S[redacted]KP001 ~]$ scp -r /home/nifiadmin/nifi-toolkit-1.23.2/ S[redacted]P003:/home/nifiadmin
nifiadmin@s[redacted]p003's password:
cli.bat                                                                100% 2615     2.9KB/s   00:00
cli.sh                                                                 100% 4210    84.7KB/s   00:00
encrypt-config.bat                                                     100% 1514     9.2KB/s   00:00
```

    b. Execute the following commands on all the nodes to encrypt the keys in nifi.properties file:

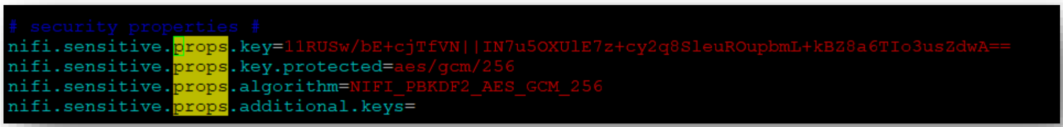
```
/home/nifiadmin/nifi-toolkit-1.23.2/bin/encrypt-config.sh -b
/home/nifiadmin/nifi-1.23.2/conf/bootstrap.conf -k
0123456789ABCDEFFEDCBA98765432100123456789ABCDEFFEDCBA987654321
0 -n /home/nifiadmin/nifi-1.23.2/conf/nifi.properties
```



```
[nifiadmin@S[redacted]P001 ~]$ vim /home/nifiadmin/nifi-1.23.2/conf/nifi.properties
[nifiadmin@S[redacted]P001 ~]$ /home/nifiadmin/nifi-toolkit-1.23.2/bin/encrypt-config.sh -b /home/nifiadmin/nifi-1.23.2/con
f/bootstrap.conf -k 012345[redacted]CDEFFEDCBA9[redacted]321[redacted]FEDCBA9[redacted]210 -n /home/nifiadmin/nifi-1.23.2/conf/n
ifi.properties
encrypt-config.sh: JAVA_HOME not set; results may vary
[main] WARN org.apache.nifi.properties.ConfigEncryptionTool - The source nifi.properties and destination nifi.properties a
re identical [/home/nifiadmin/nifi-1.23.2/conf/nifi.properties] so the original will be overwritten
[main] WARN org.apache.nifi.properties.AbstractBootstrapPropertiesLoader - System Property [nifi.properties.file.path] not
 found: Using Relative Path [conf/nifi.properties]
[main] INFO org.apache.nifi.properties.NiFiPropertiesLoader - Loading Application Properties [/home/nifiadmin/nifi-1.23.2/
conf/nifi.properties]
[main] INFO org.apache.nifi.properties.NiFiPropertiesLoader - Loading Application Properties [/home/nifiadmin/nifi-1.23.2/
conf/nifi.properties]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Loaded NiFiProperties instance with 205 properties
[main] INFO com.azure.core.implementation.ReflectionUtils - Unable to create MethodHandles to use Java 9+ MethodHandles.pr
ivateLookupIn. Will attempt to fallback to using the package-private constructor.
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Protected [nifi.security.keyPasswd] using [aes/gcm] ->    kV
f89JrGor+aae8X||[redacted]MiUS7OREyPQ6oK+BO4VfE[redacted]vAFz8ChHl[redacted]8vs4b[redacted]RFA=
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Updated protection key [nifi.security.keyPasswd.protected]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Protected [nifi.security.keystorePasswd] using [aes/gcm] ->
hQItOBq8vCeemUj+||KyZDe7Un68[redacted]tGcO4ATYaUx04T7AmJL[redacted]pMPj1F4JbZ7[redacted]12mo6Y4I=
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Updated protection key [nifi.security.keystorePasswd.protect
ed]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Protected [nifi.security.truststorePasswd] using [aes/gcm] -
>       KJxaBxfVuMb++HiZ||rTFo93lmeJS[redacted]QPWnQNuIUm/yXdXe[redacted]70XXRho3fK[redacted]LnqyWw=
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Updated protection key [nifi.security.truststorePasswd.prote
cted]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Protected [nifi.sensitive.props.key] using [aes/gcm] ->   11
RUSw/bE+cjTfVN||I[redacted]+cy2q8SleuROupbm[redacted]sZdwA==
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Updated protection key [nifi.sensitive.props.key.protected]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Final result: 208 keys including 4 protected keys
[nifiadmin@SVALIAPELKP001 ~]$
```

Ensure that the keys are in plain text before running the encryption script.

After running encryption script, keys are protected as displayed in the following image:

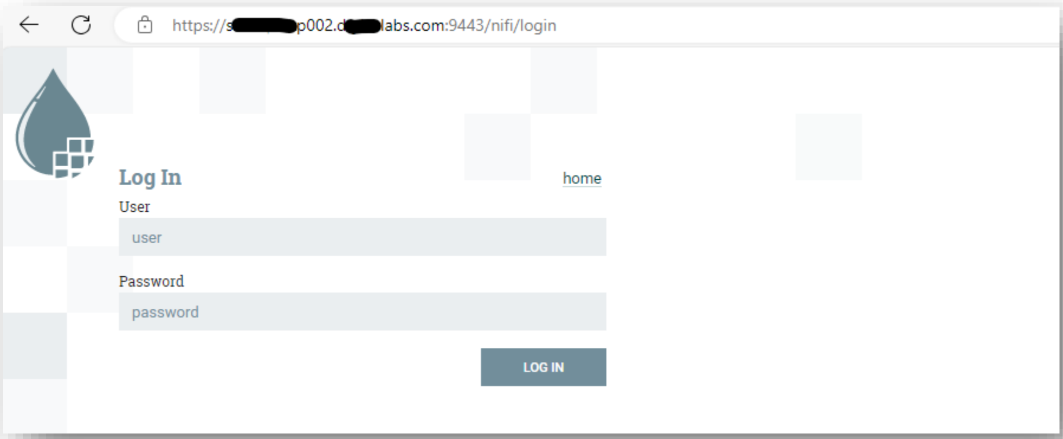


```
# security properties #
nifi.sensitive.props.key=11RUSw/bE+cjTfVN||IN7u5OXUlE7z+cy2q8SleuROupbmL+kBZ8a6TIo3usZdwA==
nifi.sensitive.props.key.protected=aes/gcm/256
nifi.sensitive.props.algorithm=NIFI_PBKDF2_AES_GCM_256
nifi.sensitive.props.additional.keys=
```

33. Start the firewalld services on all the nodes to make the connection pass between servers.
34. Run below command on all the nodes to start Nifi Service:

```
sudo service nifi start
```

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



https://[redacted]p002.d[redacted]labs.com:9443/nifi/login

Figure 3 – 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

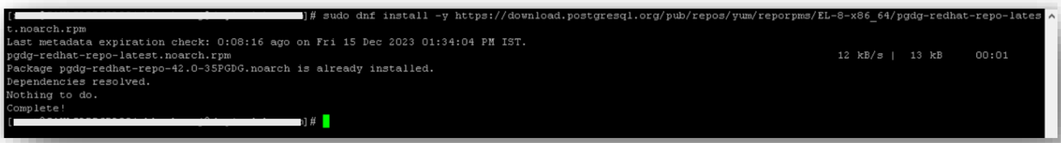
2.7.1.1 Installation Steps

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.

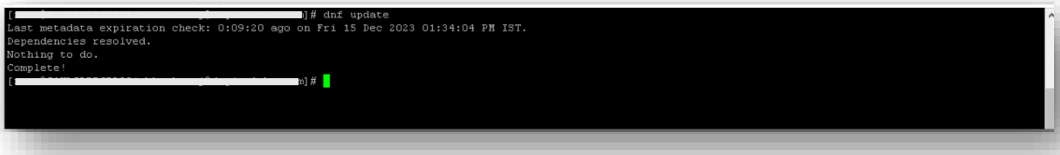
```
#  sudo dnf install -y
https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-
x86_64/pgdg-redhat-repo-latest.noarch.rpm
```



```
[[redacted]]# sudo dnf install -y https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-x86_64/pgdg-redhat-repo-lates
t.noarch.rpm
Last metadata expiration check: 0:08:16 ago on Fri 15 Dec 2023 01:34:04 PM IST.
pgdg-redhat-repo-latest.noarch.rpm                                                           12 kB/s |  13 kB     00:01
Package pgdg-redhat-repo-42.0-35PGDG.noarch is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[[redacted]]# 
```

Figure 4 – DNF install

```
# dnf update
```



```
[[redacted]]# dnf update
Last metadata expiration check: 0:09:20 ago on Fri 15 Dec 2023 01:34:04 PM IST.
Dependencies resolved.
Nothing to do.
Complete!
[[redacted]m]# 
```

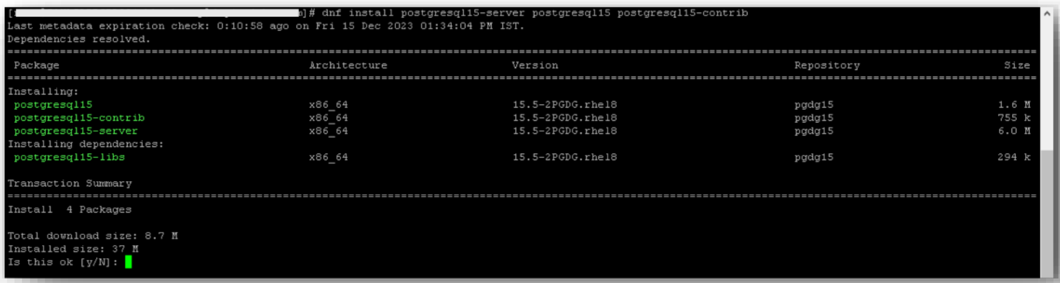
Figure 5 – DNF Update

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

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

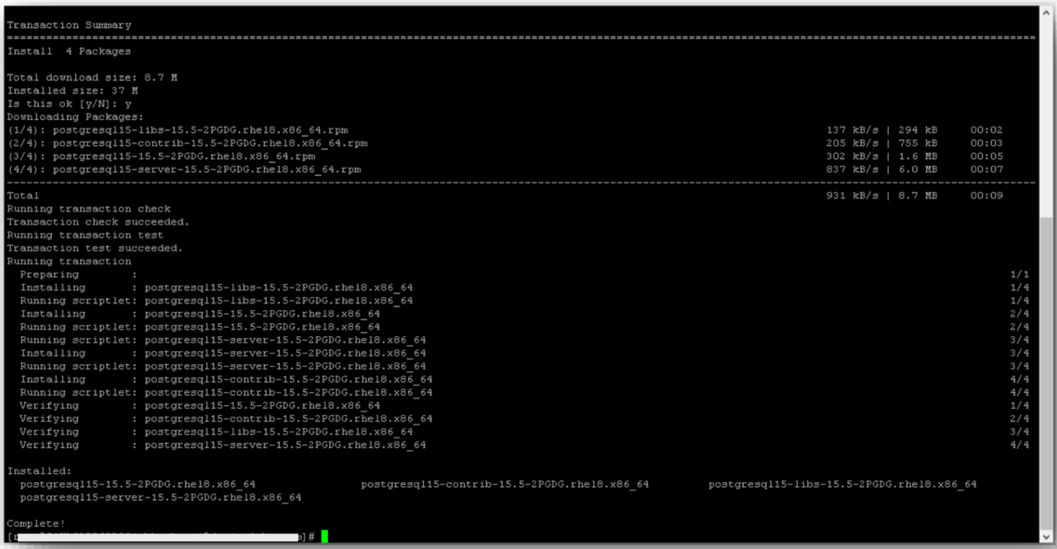
d. We can now proceed with the installation of the PostgreSQL15 database server.

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



```
[[redacted]n]# dnf install postgresql15-server postgresql15 postgresql15-contrib
Last metadata expiration check: 0:10:58 ago on Fri 15 Dec 2023 01:34:04 PM IST.
Dependencies resolved.
==========================================================================================================================
 Package                          Architecture               Version                          Repository            Size
==========================================================================================================================
Installing:
 postgresql15                     x86_64                     15.5-2PGDG.rhel8                 pgdg15               1.6 M
 postgresql15-contrib             x86_64                     15.5-2PGDG.rhel8                 pgdg15               755 k
 postgresql15-server              x86_64                     15.5-2PGDG.rhel8                 pgdg15               6.0 M
Installing dependencies:
 postgresql15-libs                x86_64                     15.5-2PGDG.rhel8                 pgdg15               294 k

Transaction Summary
==========================================================================================================================
Install  4 Packages

Total download size: 8.7 M
Installed size: 37 M
Is this ok [y/N]: 
```

Figure 6 – installing PostgreSQL15



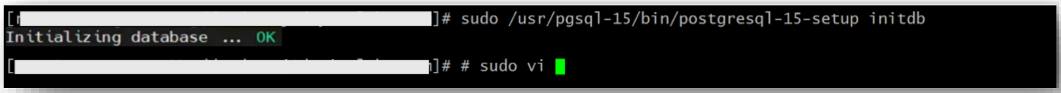
```
Transaction Summary
================================================================================================================================================================
Install  4 Packages

Total download size: 8.7 M
Installed size: 37 M
Is this ok [y/N]: y
Downloading Packages:
(1/4): postgresql15-libs-15.5-2PGDG.rhel8.x86_64.rpm                                                                       137 kB/s | 294 kB     00:02
(2/4): postgresql15-contrib-15.5-2PGDG.rhel8.x86_64.rpm                                                                    205 kB/s | 755 kB     00:03
(3/4): postgresql15-15.5-2PGDG.rhel8.x86_64.rpm                                                                            302 kB/s | 1.6 MB     00:05
(4/4): postgresql15-server-15.5-2PGDG.rhel8.x86_64.rpm                                                                     837 kB/s | 6.0 MB     00:07
----------------------------------------------------------------------------------------------------------------------------------------------------------------
Total                                                                                                                      931 kB/s | 8.7 MB     00:09
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                                                      1/1
  Installing       : postgresql15-libs-15.5-2PGDG.rhel8.x86_64                                                                                            1/4
  Running scriptlet: postgresql15-libs-15.5-2PGDG.rhel8.x86_64                                                                                            1/4
  Installing       : postgresql15-15.5-2PGDG.rhel8.x86_64                                                                                                 2/4
  Running scriptlet: postgresql15-15.5-2PGDG.rhel8.x86_64                                                                                                 2/4
  Running scriptlet: postgresql15-server-15.5-2PGDG.rhel8.x86_64                                                                                          3/4
  Installing       : postgresql15-server-15.5-2PGDG.rhel8.x86_64                                                                                          3/4
  Running scriptlet: postgresql15-server-15.5-2PGDG.rhel8.x86_64                                                                                          3/4
  Installing       : postgresql15-contrib-15.5-2PGDG.rhel8.x86_64                                                                                         4/4
  Running scriptlet: postgresql15-contrib-15.5-2PGDG.rhel8.x86_64                                                                                         4/4
  Verifying        : postgresql15-15.5-2PGDG.rhel8.x86_64                                                                                                 1/4
  Verifying        : postgresql15-contrib-15.5-2PGDG.rhel8.x86_64                                                                                         2/4
  Verifying        : postgresql15-libs-15.5-2PGDG.rhel8.x86_64                                                                                            3/4
  Verifying        : postgresql15-server-15.5-2PGDG.rhel8.x86_64                                                                                          4/4

Installed:
  postgresql15-15.5-2PGDG.rhel8.x86_64                 postgresql15-contrib-15.5-2PGDG.rhel8.x86_64         postgresql15-libs-15.5-2PGDG.rhel8.x86_64
  postgresql15-server-15.5-2PGDG.rhel8.x86_64

Complete!
[[redacted]]#
```

Figure 7– 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
```

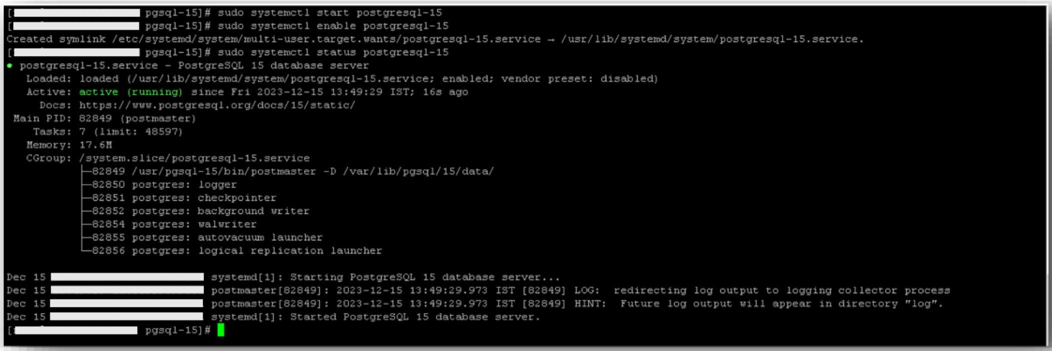


```
[[redacted]]# sudo /usr/pgsql-15/bin/postgresql-15-setup initdb
Initializing database ... OK
[[redacted]]# # sudo vi
```

Figure 8 - 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
```



```
[[redacted] pgsql-15]# sudo systemctl start postgresql-15
[[redacted] pgsql-15]# sudo systemctl enable postgresql-15
Created symlink /etc/systemd/system/multi-user.target.wants/postgresql-15.service → /usr/lib/systemd/system/postgresql-15.service.
[[redacted] pgsql-15]# sudo systemctl status postgresql-15
● postgresql-15.service - PostgreSQL 15 database server
   Loaded: loaded (/usr/lib/systemd/system/postgresql-15.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2023-12-15 13:49:29 IST; 16s ago
     Docs: https://www.postgresql.org/docs/15/static/
 Main PID: 82849 (postmaster)
    Tasks: 7 (limit: 48597)
   Memory: 17.6M
   CGroup: /system.slice/postgresql-15.service
           ├─82849 /usr/pgsql-15/bin/postmaster -D /var/lib/pgsql/15/data/
           ├─82850 postgres: logger
           ├─82851 postgres: checkpointer
           ├─82852 postgres: background writer
           ├─82854 postgres: walwriter
           ├─82855 postgres: autovacuum launcher
           └─82856 postgres: logical replication launcher

Dec 15 [redacted] systemd[1]: Starting PostgreSQL 15 database server...
Dec 15 [redacted] postmaster[82849]: 2023-12-15 13:49:29.973 IST [82849] LOG:  redirecting log output to logging collector process
Dec 15 [redacted] postmaster[82849]: 2023-12-15 13:49:29.973 IST [82849] HINT:  Future log output will appear in directory "log".
Dec 15 [redacted] systemd[1]: Started PostgreSQL 15 database server.
[[redacted] pgsql-15]#
```

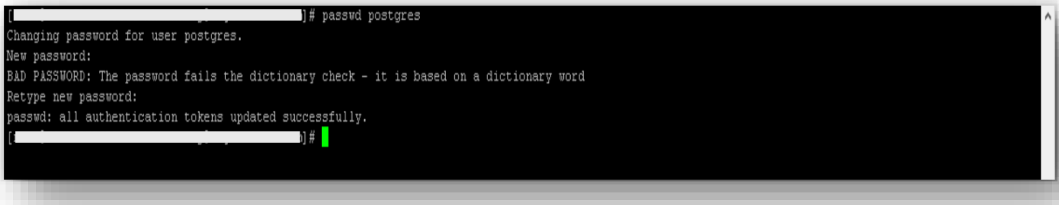
Figure 9 – Systemctl Status

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
```

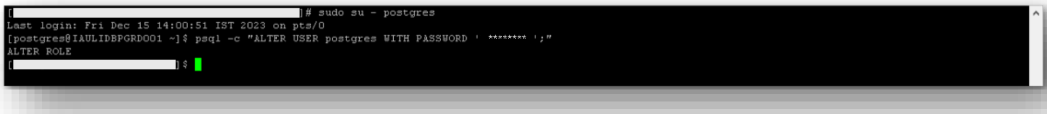


```
[[redacted]]# passwd postgres
Changing password for user postgres.
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
Retype new password:
passwd: all authentication tokens updated successfully.
[[redacted]]#
```

Figure 10 - Create Password

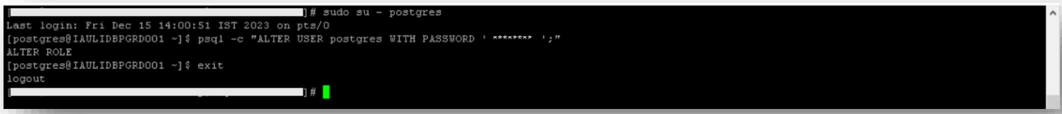
b. Switch to the postgres system user account and secure the PostgreSQL administrative database user account by creating a password for it (remember to set a strong and secure password).

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



```
[[redacted]]# sudo su - postgres
Last login: Fri Dec 15 14:00:51 IST 2023 on pts/0
[postgres@IAULIDBPGRD001 ~]$ psql -c "ALTER USER postgres WITH PASSWORD ' ******** ';"
ALTER ROLE
[[redacted]]$
```

Figure 11 – Alter Role



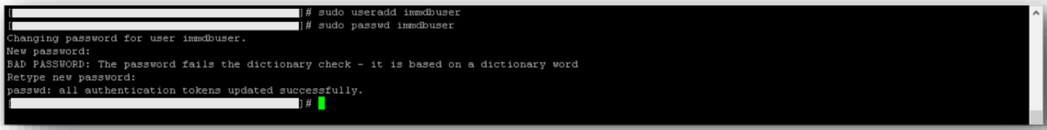
```
[[redacted]]# sudo su - postgres
Last login: Fri Dec 15 14:00:51 IST 2023 on pts/0
[postgres@IAULIDBPGRD001 ~]$ psql -c "ALTER USER postgres WITH PASSWORD ' ******** ';"
ALTER ROLE
[postgres@IAULIDBPGRD001 ~]$ exit
logout
[[redacted]]#
```

Figure 12 – Alter Role (Cont.)

c. Create a New PostgreSQL "immdbuser" User Account

i. First create Linux User Account

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



```
[[redacted]]# sudo useradd immdbuser
[[redacted]]# sudo passwd immdbuser
Changing password for user immdbuser.
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
Retype new password:
passwd: all authentication tokens updated successfully.
[[redacted]]#
```

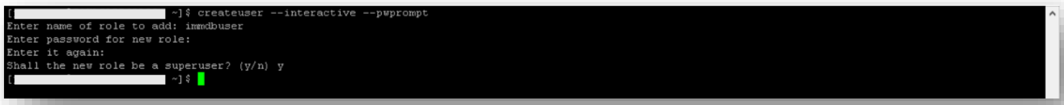
Figure 13 – Create User Account

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
```

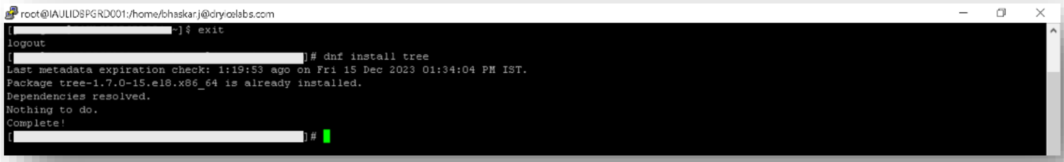


```
[[redacted] ~]$ createuser --interactive --pwprompt
Enter name of role to add: immdbuser
Enter password for new role:
Enter it again:
Shall the new role be a superuser? (y/n) y
[[redacted] ~]$
```

Figure 14 – 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/
```



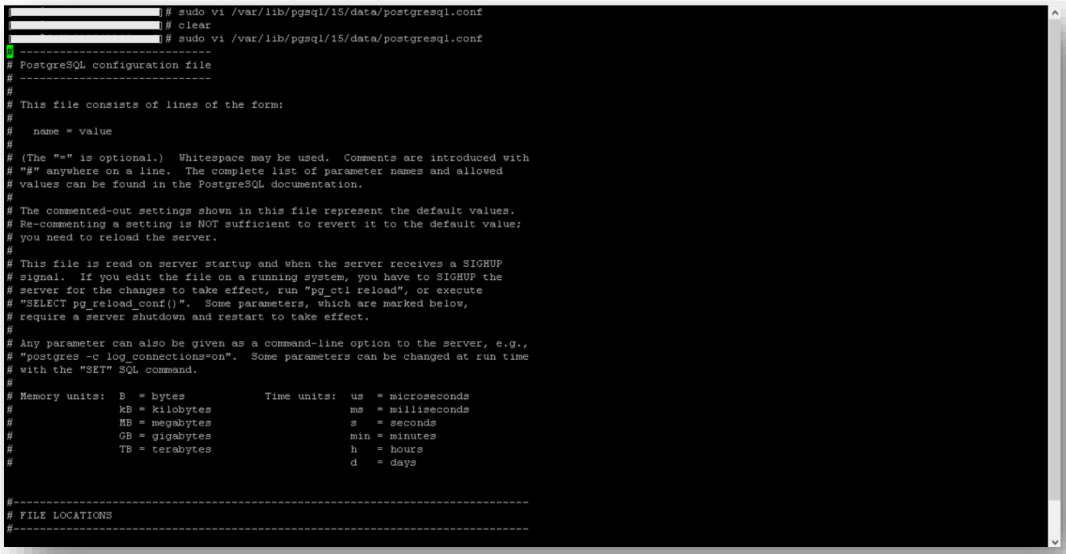
```
root@IAULIDBPGRD001:/home/bhaskar.j@dryicelabs.com
[[redacted]~]$ exit
logout
[[redacted]]# dnf install tree
Last metadata expiration check: 1:19:53 ago on Fri 15 Dec 2023 01:34:04 PM IST.
Package tree-1.7.0-15.el8.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[[redacted]]#
```

Figure 15 – 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
```

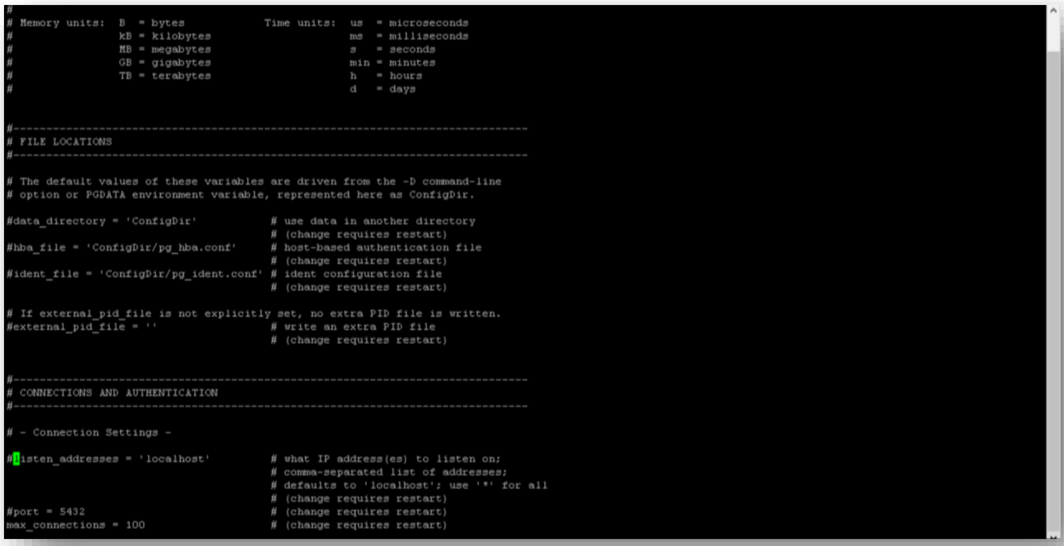


```
[[redacted]]# sudo vi /var/lib/pgsql/15/data/postgresql.conf
[[redacted]]# clear
[[redacted]]# sudo vi /var/lib/pgsql/15/data/postgresql.conf
# -----------------------------
# PostgreSQL configuration file
# -----------------------------
#
# This file consists of lines of the form:
#
#   name = value
#
# (The "=" is optional.)  Whitespace may be used.  Comments are introduced with
# "#" anywhere on a line.  The complete list of parameter names and allowed
# values can be found in the PostgreSQL documentation.
#
# The commented-out settings shown in this file represent the default values.
# Re-commenting a setting is NOT sufficient to revert it to the default value;
# you need to reload the server.
#
# This file is read on server startup and when the server receives a SIGHUP
# signal.  If you edit the file on a running system, you have to SIGHUP the
# server for the changes to take effect, run "pg_ctl reload", or execute
# "SELECT pg_reload_conf()".  Some parameters, which are marked below,
# require a server shutdown and restart to take effect.
#
# Any parameter can also be given as a command-line option to the server, e.g.,
# "postgres -c log_connections=on".  Some parameters can be changed at run time
# with the "SET" SQL command.
#
# Memory units:  B  = bytes            Time units:  us  = microseconds
#                kB = kilobytes                     ms  = milliseconds
#                MB = megabytes                     s   = seconds
#                GB = gigabytes                     min = minutes
#                TB = terabytes                     h   = hours
#                                                   d   = days


#------------------------------------------------------------------------------
# FILE LOCATIONS
#------------------------------------------------------------------------------
```

Figure 16 – VI postgresql.conf

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

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

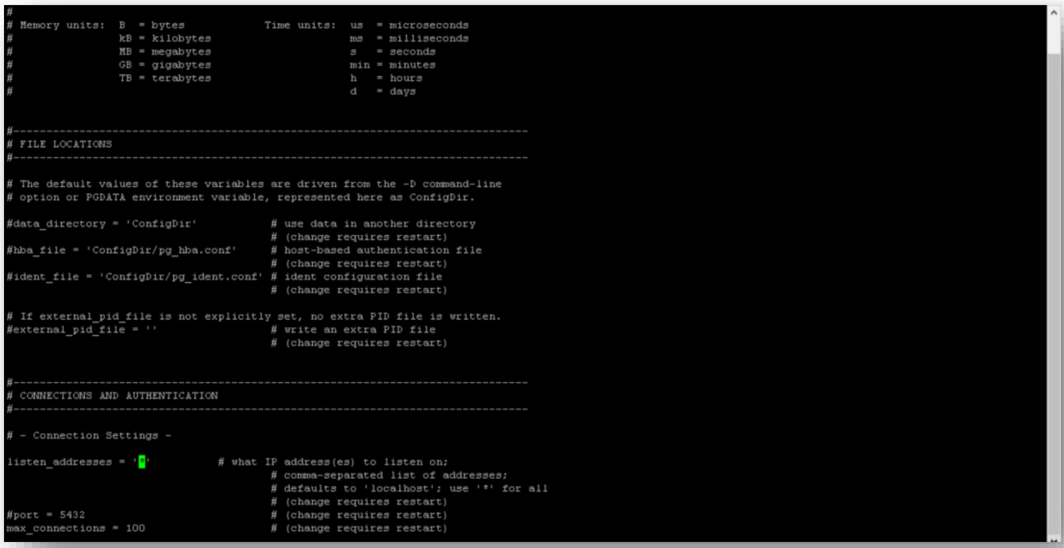


```
#
# Memory units:  B  = bytes            Time units:  us  = microseconds
#                kB = kilobytes                     ms  = milliseconds
#                MB = megabytes                     s   = seconds
#                GB = gigabytes                     min = minutes
#                TB = terabytes                     h   = hours
#                                                   d   = days


#------------------------------------------------------------------------------
# FILE LOCATIONS
#------------------------------------------------------------------------------

# The default values of these variables are driven from the -D command-line
# option or PGDATA environment variable, represented here as ConfigDir.

#data_directory = 'ConfigDir'           # use data in another directory
                                        # (change requires restart)
#hba_file = 'ConfigDir/pg_hba.conf'     # host-based authentication file
                                        # (change requires restart)
#ident_file = 'ConfigDir/pg_ident.conf' # ident configuration file
                                        # (change requires restart)

# If external_pid_file is not explicitly set, no extra PID file is written.
#external_pid_file = ''                 # write an extra PID file
                                        # (change requires restart)


#------------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------

# - Connection Settings -

#listen_addresses = 'localhost'         # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for all
                                        # (change requires restart)
#port = 5432                            # (change requires restart)
max_connections = 100                   # (change requires restart)
```

Figure 17 – Connection Setting

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

```
listen_addresses = '*'
```



```
#
# Memory units:  B  = bytes            Time units:  us  = microseconds
#                kB = kilobytes                     ms  = milliseconds
#                MB = megabytes                     s   = seconds
#                GB = gigabytes                     min = minutes
#                TB = terabytes                     h   = hours
#                                                   d   = days


#------------------------------------------------------------------------------
# FILE LOCATIONS
#------------------------------------------------------------------------------

# The default values of these variables are driven from the -D command-line
# option or PGDATA environment variable, represented here as ConfigDir.

#data_directory = 'ConfigDir'           # use data in another directory
                                        # (change requires restart)
#hba_file = 'ConfigDir/pg_hba.conf'     # host-based authentication file
                                        # (change requires restart)
#ident_file = 'ConfigDir/pg_ident.conf' # ident configuration file
                                        # (change requires restart)

# If external_pid_file is not explicitly set, no extra PID file is written.
#external_pid_file = ''                 # write an extra PID file
                                        # (change requires restart)


#------------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------

# - Connection Settings -

listen_addresses = '*'          # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for all
                                        # (change requires restart)
#port = 5432                            # (change requires restart)
max_connections = 100                   # (change requires restart)
```

Figure 18 – Connection Setting (Cont.)

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

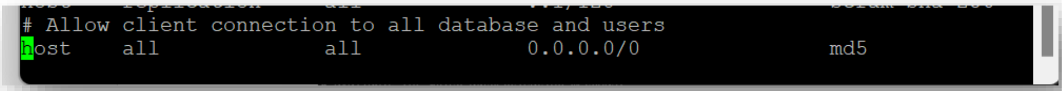
```
# sudo systemctl restart postgresql-15
```

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.

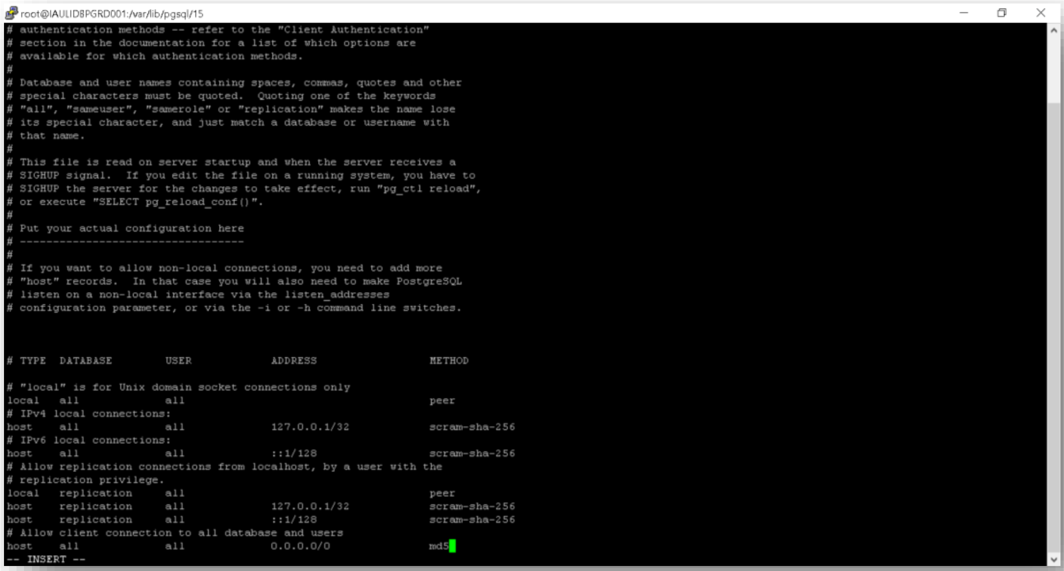
```
# vi /var/lib/pgsql/15/data/pg_hba.conf
```

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



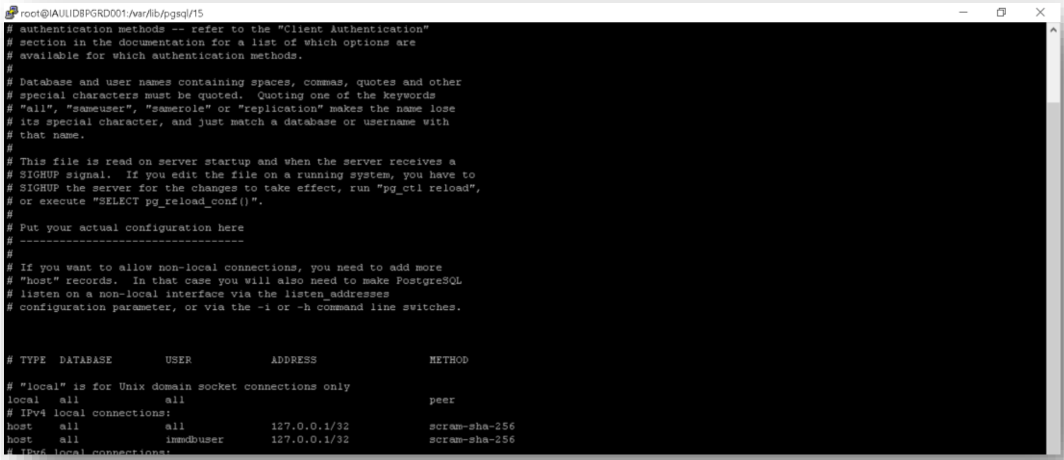
```
# Allow client connection to all database and users
host    all             all             0.0.0.0/0               md5
```

Figure 19 – Security Setting



```
# authentication methods -- refer to the "Client Authentication"
# section in the documentation for a list of which options are
# available for which authentication methods.
#
# Database and user names containing spaces, commas, quotes and other
# special characters must be quoted.  Quoting one of the keywords
# "all", "sameuser", "samerole" or "replication" makes the name lose
# its special character, and just match a database or username with
# that name.
#
# This file is read on server startup and when the server receives a
# SIGHUP signal.  If you edit the file on a running system, you have to
# SIGHUP the server for the changes to take effect, run "pg_ctl reload",
# or execute "SELECT pg_reload_conf()".
#
# Put your actual configuration here
# ----------------------------------
#
# If you want to allow non-local connections, you need to add more
# "host" records.  In that case you will also need to make PostgreSQL
# listen on a non-local interface via the listen_addresses
# configuration parameter, or via the -i or -h command line switches.



# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     peer
# IPv4 local connections:
host    all             all             127.0.0.1/32            scram-sha-256
# IPv6 local connections:
host    all             all             ::1/128                 scram-sha-256
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication     all                                     peer
host    replication     all             127.0.0.1/32            scram-sha-256
host    replication     all             ::1/128                 scram-sha-256
# Allow client connection to all database and users
host    all             all             0.0.0.0/0               md5
-- INSERT --
```

Figure 20 – Security Setting (Cont.)



```
# authentication methods -- refer to the "Client Authentication"
# section in the documentation for a list of which options are
# available for which authentication methods.
#
# Database and user names containing spaces, commas, quotes and other
# special characters must be quoted.  Quoting one of the keywords
# "all", "sameuser", "samerole" or "replication" makes the name lose
# its special character, and just match a database or username with
# that name.
#
# This file is read on server startup and when the server receives a
# SIGHUP signal.  If you edit the file on a running system, you have to
# SIGHUP the server for the changes to take effect, run "pg_ctl reload",
# or execute "SELECT pg_reload_conf()".
#
# Put your actual configuration here
# ----------------------------------
#
# If you want to allow non-local connections, you need to add more
# "host" records.  In that case you will also need to make PostgreSQL
# listen on a non-local interface via the listen_addresses
# configuration parameter, or via the -i or -h command line switches.



# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     peer
# IPv4 local connections:
host    all             all             127.0.0.1/32            scram-sha-256
host    all             immdbuser       127.0.0.1/32            scram-sha-256
# IPv6 local connections:
```

Figure 21 – 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

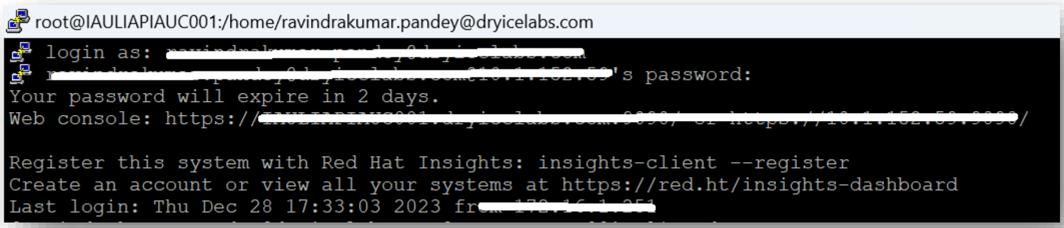
This section explains how to install IMM components using the installer on Linux server or standalone machine. The installer can be further used for deployment of web server, application server and , database server.

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:

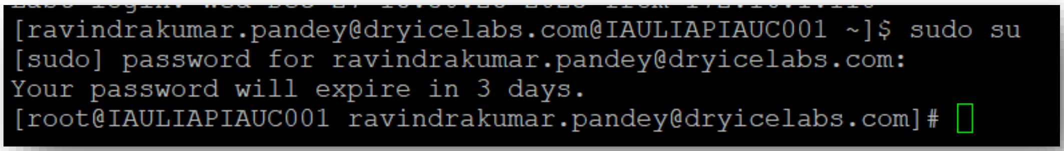
1. Open putty and Login with valid credentials to the targeted server where you want to install IMM Application.



```
root@IAULIAPIAUC001:/home/ravindrakumar.pandey@dryicelabs.com
login as: [redacted]
[redacted]'s password:
Your password will expire in 2 days.
Web console: https://[redacted]/

Register this system with Red Hat Insights: insights-client --register
Create an account or view all your systems at https://red.ht/insights-dashboard
Last login: Thu Dec 28 17:33:03 2023 fr[redacted]
```

Figure 22 – Login to Server.

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



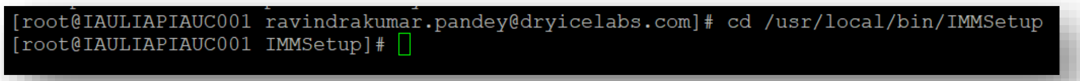
```
[ravindrakumar.pandey@dryicelabs.com@IAULIAPIAUC001 ~]$ sudo su
[sudo] password for ravindrakumar.pandey@dryicelabs.com:
Your password will expire in 3 days.
[root@IAULIAPIAUC001 ravindrakumar.pandey@dryicelabs.com]#
```

Figure 23 – 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



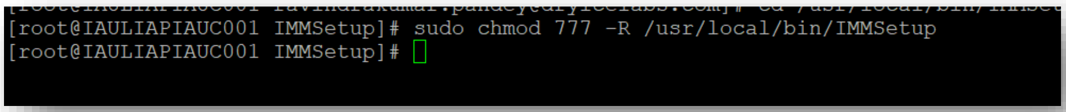
```
[root@IAULIAPIAUC001 ravindrakumar.pandey@dryicelabs.com]# cd /usr/local/bin/IMMSetup
[root@IAULIAPIAUC001 IMMSetup]#
```

Figure 24 – 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

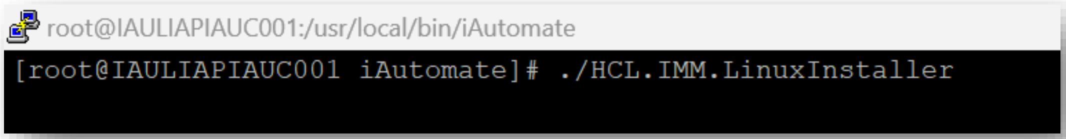


```
[root@IAULIAPIAUC001 IMMSetup]# sudo chmod 777 -R /usr/local/bin/IMMSetup
[root@IAULIAPIAUC001 IMMSetup]#
```

Figure 25 – Provide access to file.

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

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



root@IAULIAPIAUC001:/usr/local/bin/iAutomate

```
[root@IAULIAPIAUC001 iAutomate]# ./HCL.IMM.LinuxInstaller
```

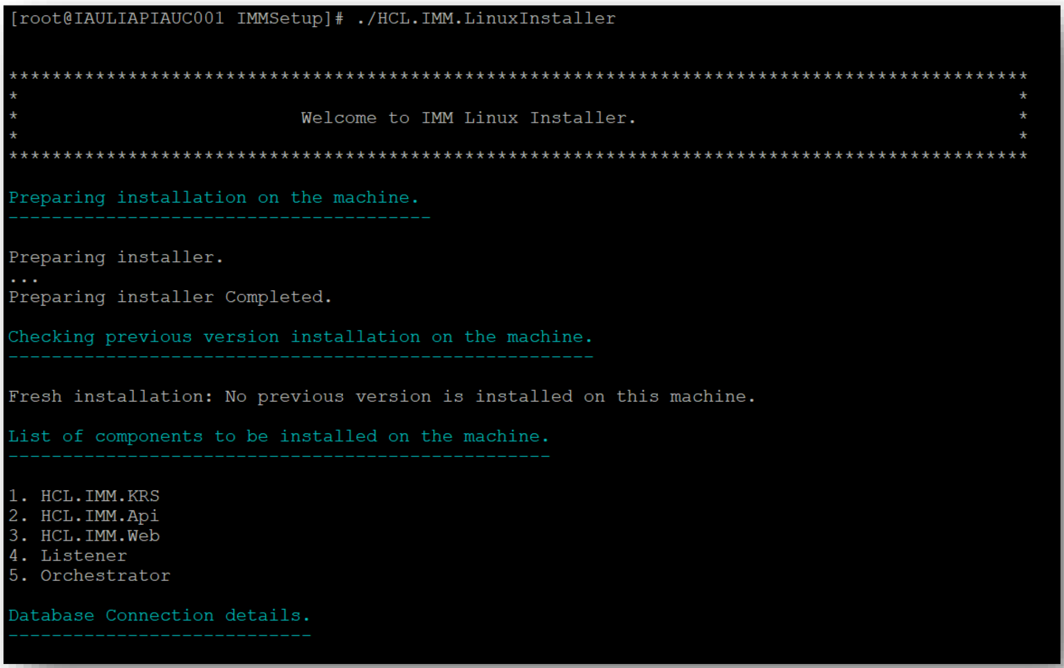
Figure 26 - Run the Installer

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:

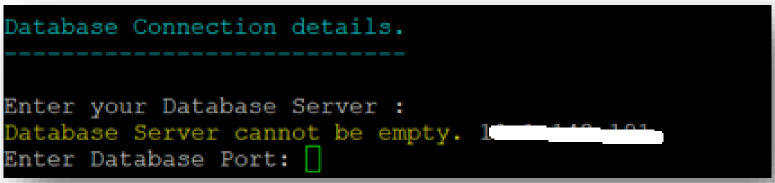
1. On running the Installer, the following page appears.



```
[root@IAULIAPIAUC001 IMMSetup]# ./HCL.IMM.LinuxInstaller

*********************************************************************************************
*                                                                                           *
*                          Welcome to IMM Linux Installer.                                  *
*                                                                                           *
*********************************************************************************************

Preparing installation on the machine.
---------------------------------------

Preparing installer.
...
Preparing installer Completed.

Checking previous version installation on the machine.
------------------------------------------------------

Fresh installation: No previous version is installed on this machine.

List of components to be installed on the machine.
--------------------------------------------------

1. HCL.IMM.KRS
2. HCL.IMM.Api
3. HCL.IMM.Web
4. Listener
5. Orchestrator

Database Connection details.
----------------------------
```

Figure 27 - 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 27 - 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.

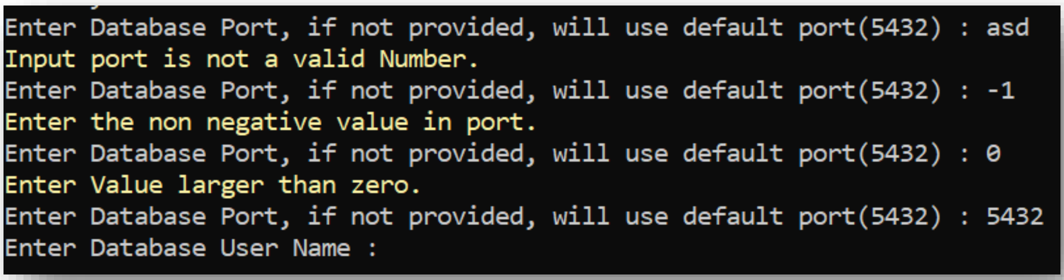


```
Database Connection details.
----------------------------

Enter your Database Server :
Database Server cannot be empty. 1[redacted]
Enter Database Port:
```

Figure 28 – 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.

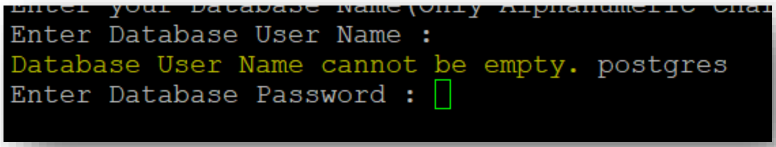


```
Enter Database Port, if not provided, will use default port(5432) : asd
Input port is not a valid Number.
Enter Database Port, if not provided, will use default port(5432) : -1
Enter the non negative value in port.
Enter Database Port, if not provided, will use default port(5432) : 0
Enter Value larger than zero.
Enter Database Port, if not provided, will use default port(5432) : 5432
Enter Database User Name :
```

Figure 29 – Database Connection Details, Database Port.

5. Input the **Database Username**.

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

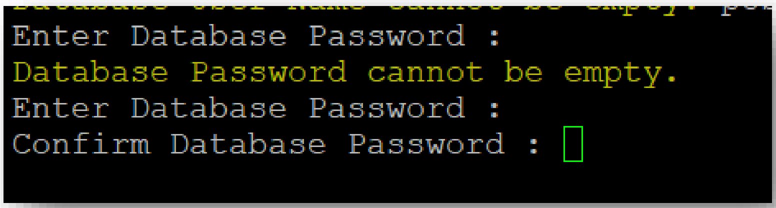


```
Enter your Database Name(Only Alphanumeric Char
Enter Database User Name :
Database User Name cannot be empty. postgres
Enter Database Password :
```

Figure 30 – 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.



```
Database User Name cannot be empty. pos
Enter Database Password :
Database Password cannot be empty.
Enter Database Password :
Confirm Database Password :
```

Figure 31 – 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.

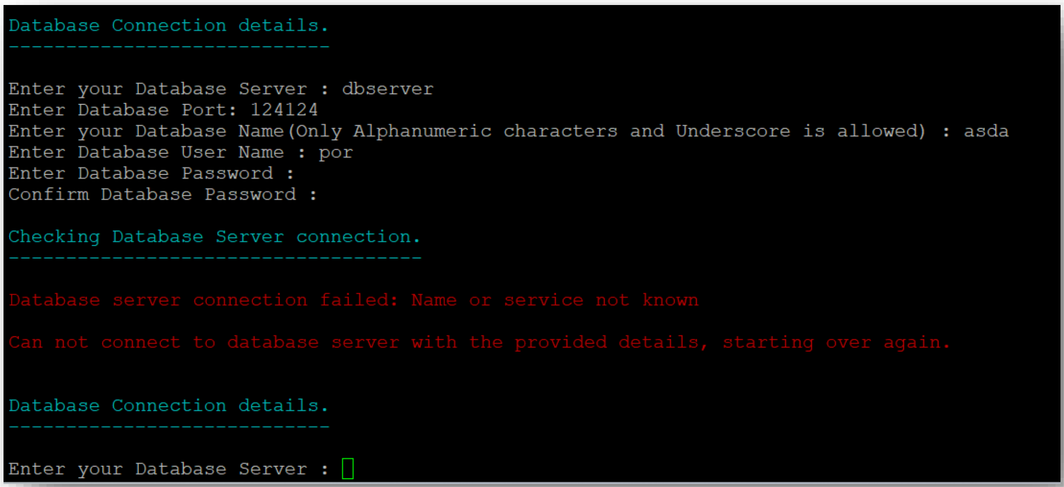

```
Database Connection details.
----------------------------

Enter your Database Server : dbserver
Enter Database Port: 124124
Enter your Database Name(Only Alphanumeric characters and Underscore is allowed) : asda
Enter Database User Name : por
Enter Database Password :
Confirm Database Password :

Checking Database Server connection.
------------------------------------

Database server connection failed: Name or service not known

Can not connect to database server with the provided details, starting over again.


Database Connection details.
----------------------------

Enter your Database Server :
```

Figure 32 – Database Connection Validation – Connection Failed.

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

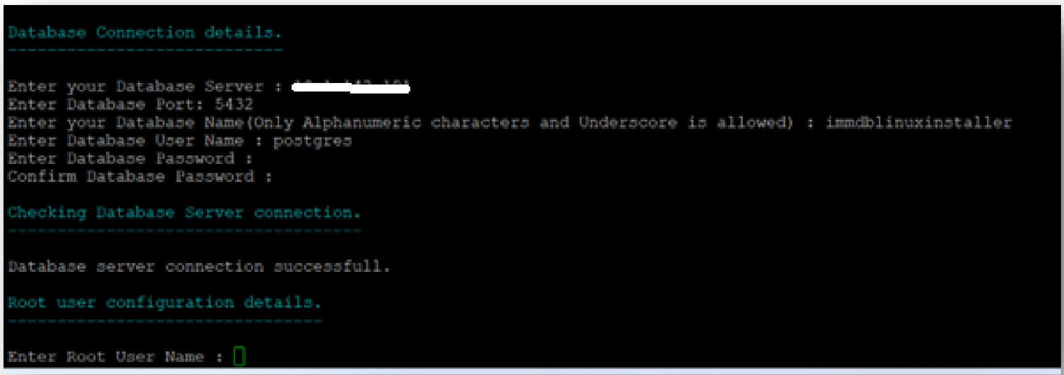

```
Database Connection details.
----------------------------

Enter your Database Server : [redacted]
Enter Database Port: 5432
Enter your Database Name(Only Alphanumeric characters and Underscore is allowed) : immdblinuxinstaller
Enter Database User Name : postgres
Enter Database Password :
Confirm Database Password :

Checking Database Server connection.
------------------------------------

Database server connection successfull.

Root user configuration details.
--------------------------------

Enter Root User Name :
```

Figure 33 – 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.

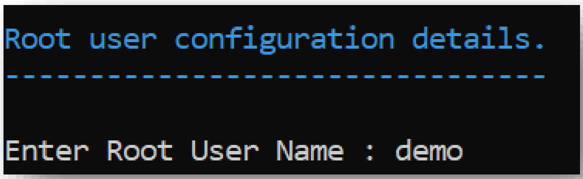
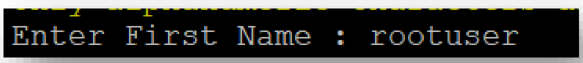

```
Root user configuration details.
--------------------------------

Enter Root User Name : demo
```

Figure 34 – 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.

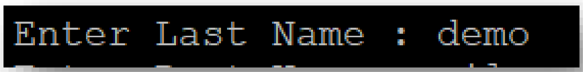


```
Enter First Name : rootuser
```

Figure 35 – 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.

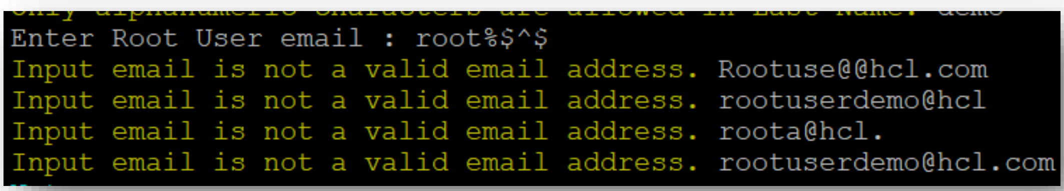


```
Enter Last Name : demo
```

Figure 36 – Root User details – Last Name

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

    Validation: It has valid email input validation.



```
Enter Root User email : root%$^$
Input email is not a valid email address. Rootuse@@hcl.com
Input email is not a valid email address. rootuserdemo@hcl
Input email is not a valid email address. roota@hcl.
Input email is not a valid email address. rootuserdemo@hcl.com
```

Figure 37 – 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:

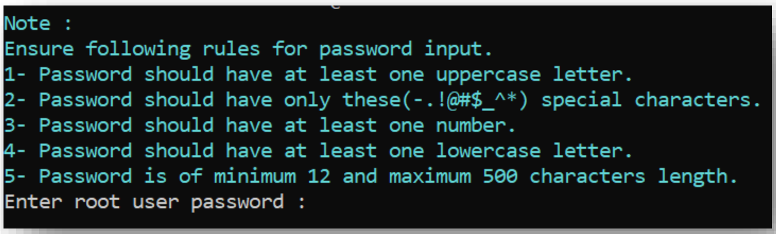
1- Password must contain at least one uppercase letter.

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.

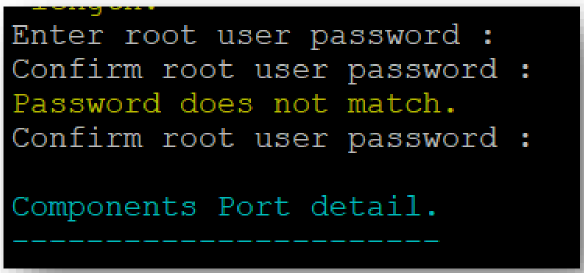


```
Note :
Ensure following rules for password input.
1- Password should have at least one uppercase letter.
2- Password should have only these(-.!@#$_^*) special characters.
3- Password should have at least one number.
4- Password should have at least one lowercase letter.
5- Password is of minimum 12 and maximum 500 characters length.
Enter root user password :
```

Figure 38 – 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.

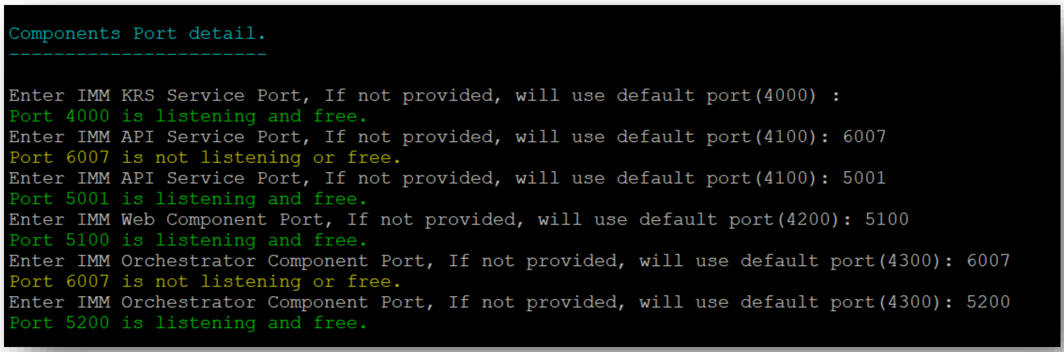


```
Enter root user password :
Confirm root user password :
Password does not match.
Confirm root user password :

Components Port detail.
------------------------
```

Figure 39 - Root User details – Confirm Root User Password.

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

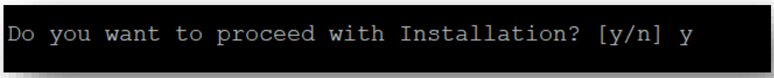
    Validation: The Installer checks for entered port for every application whether they are open - listening and not in use by any other application.



```
Components Port detail.
-----------------------

Enter IMM KRS Service Port, If not provided, will use default port(4000) :
Port 4000 is listening and free.
Enter IMM API Service Port, If not provided, will use default port(4100): 6007
Port 6007 is not listening or free.
Enter IMM API Service Port, If not provided, will use default port(4100): 5001
Port 5001 is listening and free.
Enter IMM Web Component Port, If not provided, will use default port(4200): 5100
Port 5100 is listening and free.
Enter IMM Orchestrator Component Port, If not provided, will use default port(4300): 6007
Port 6007 is not listening or free.
Enter IMM Orchestrator Component Port, If not provided, will use default port(4300): 5200
Port 5200 is listening and free.
```

Figure 40 – Input port for KRS, API, Web, and Orchestrator applications

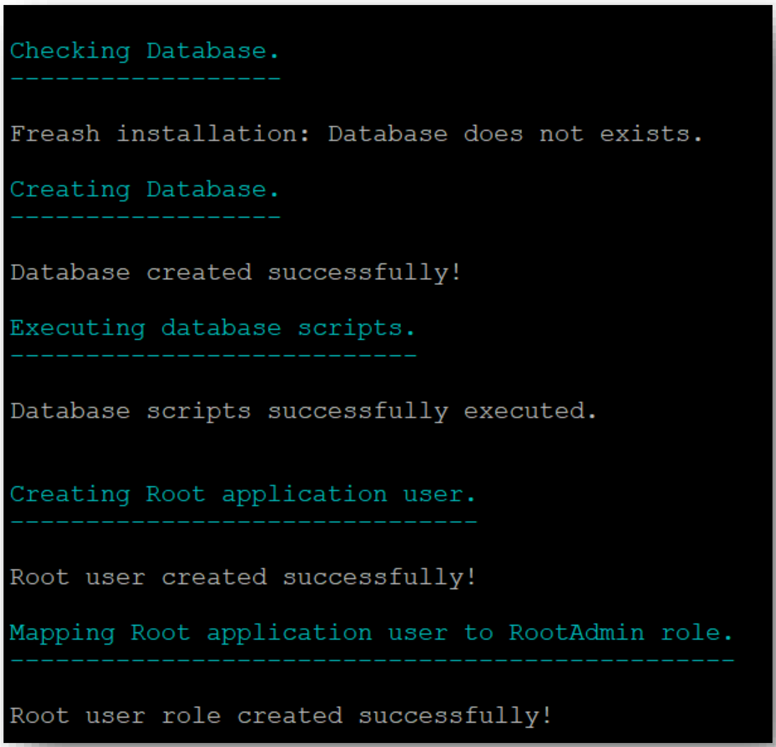
17. After capturing all the details, the installer confirms from the user if he wants to proceed with the installation.



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

Figure 41 – Installation confirmation before proceeding further.

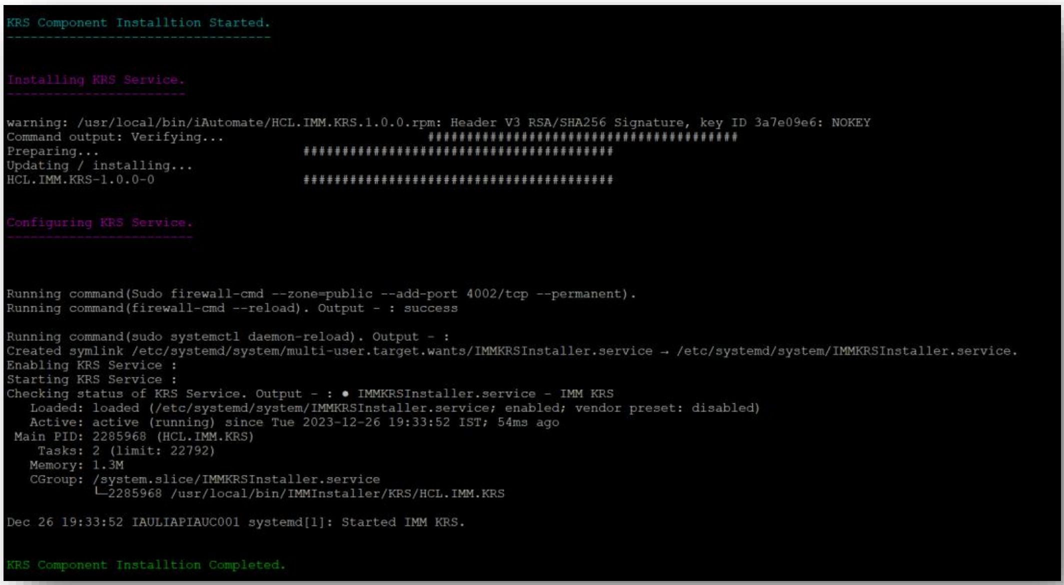
18. Installer further checks if the input database exists. If not, it creates it and runs the database scripts on it. It also creates the root user and maps this user to root admin role.



```
Checking Database.
------------------

Freash installation: Database does not exists.

Creating Database.
------------------

Database created successfully!

Executing database scripts.
---------------------------

Database scripts successfully executed.


Creating Root application user.
-------------------------------

Root user created successfully!

Mapping Root application user to RootAdmin role.
------------------------------------------------

Root user role created successfully!
```

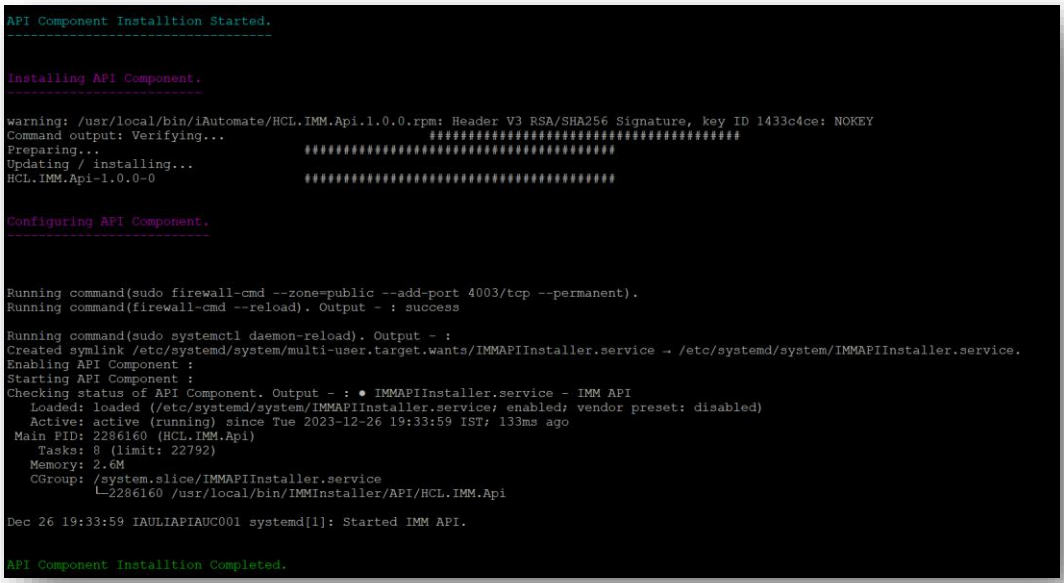
Figure 42 – Installation Database Check and Other Database Tasks

The components installation starts.



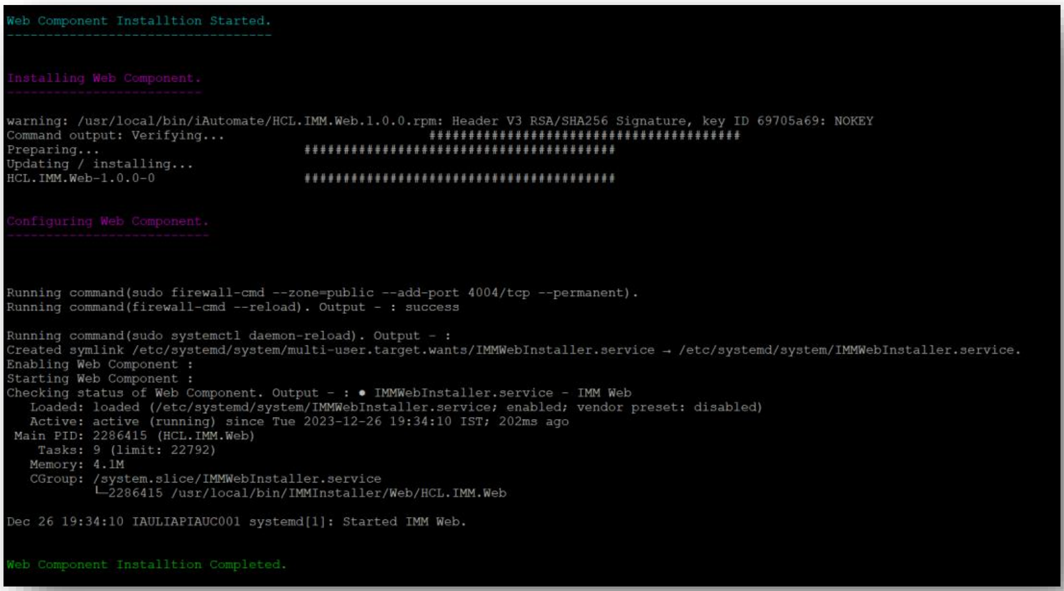
```
KRS Component Installtion Started.
----------------------------------


Installing KRS Service.
-----------------------

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


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



Running command(Sudo firewall-cmd --zone=public --add-port 4002/tcp --permanent).
Running command(firewall-cmd --reload). Output - : success

Running command(sudo systemctl daemon-reload). Output - :
Created symlink /etc/systemd/system/multi-user.target.wants/IMMKRSInstaller.service → /etc/systemd/system/IMMKRSInstaller.service.
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; 54ms ago
 Main PID: 2285968 (HCL.IMM.KRS)
    Tasks: 2 (limit: 22792)
   Memory: 1.3M
   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.


KRS Component Installtion Completed.
```

Figure 43 – KRS Component Installation



```
API Component Installtion Started.
----------------------------------


Installing API Component.
--------------------------

warning: /usr/local/bin/iAutomate/HCL.IMM.Api.1.0.0.rpm: Header V3 RSA/SHA256 Signature, key ID 1433c4ce: NOKEY
Command output: Verifying...                          ########################################
Preparing...                          ########################################
Updating / installing...
HCL.IMM.Api-1.0.0-0                   ########################################


Configuring API Component.
--------------------------



Running command(sudo firewall-cmd --zone=public --add-port 4003/tcp --permanent).
Running command(firewall-cmd --reload). Output - : success

Running command(sudo systemctl daemon-reload). Output - :
Created symlink /etc/systemd/system/multi-user.target.wants/IMMAPIInstaller.service → /etc/systemd/system/IMMAPIInstaller.service.
Enabling API Component :
Starting API Component :
Checking status of API Component. Output - : ● IMMAPIInstaller.service - IMM API
   Loaded: loaded (/etc/systemd/system/IMMAPIInstaller.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2023-12-26 19:33:59 IST; 133ms ago
 Main PID: 2286160 (HCL.IMM.Api)
    Tasks: 8 (limit: 22792)
   Memory: 2.6M
   CGroup: /system.slice/IMMAPIInstaller.service
           └─2286160 /usr/local/bin/IMMInstaller/API/HCL.IMM.Api

Dec 26 19:33:59 IAULIAPIAUC001 systemd[1]: Started IMM API.


API Component Installtion Completed.
```

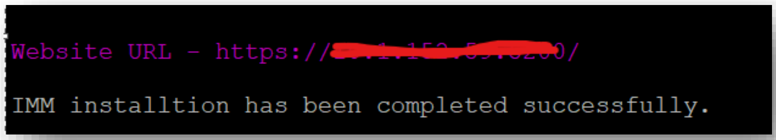
Figure 44 – API Component Installation



```
Web Component Installtion Started.
----------------------------------


Installing Web Component.
--------------------------

warning: /usr/local/bin/iAutomate/HCL.IMM.Web.1.0.0.rpm: Header V3 RSA/SHA256 Signature, key ID 69705a69: NOKEY
Command output: Verifying...                          ########################################
Preparing...                          ########################################
Updating / installing...
HCL.IMM.Web-1.0.0-0                   ########################################


Configuring Web Component.
--------------------------



Running command(sudo firewall-cmd --zone=public --add-port 4004/tcp --permanent).
Running command(firewall-cmd --reload). Output - : success

Running command(sudo systemctl daemon-reload). Output - :
Created symlink /etc/systemd/system/multi-user.target.wants/IMMWebInstaller.service → /etc/systemd/system/IMMWebInstaller.service.
Enabling Web Component :
Starting Web Component :
Checking status of Web Component. Output - : ● IMMWebInstaller.service - IMM Web
   Loaded: loaded (/etc/systemd/system/IMMWebInstaller.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2023-12-26 19:34:10 IST; 202ms ago
 Main PID: 2286415 (HCL.IMM.Web)
    Tasks: 9 (limit: 22792)
   Memory: 4.1M
   CGroup: /system.slice/IMMWebInstaller.service
           └─2286415 /usr/local/bin/IMMInstaller/Web/HCL.IMM.Web

Dec 26 19:34:10 IAULIAPIAUC001 systemd[1]: Started IMM Web.


Web Component Installtion Completed.
```

Figure 45 – Web Component installation.

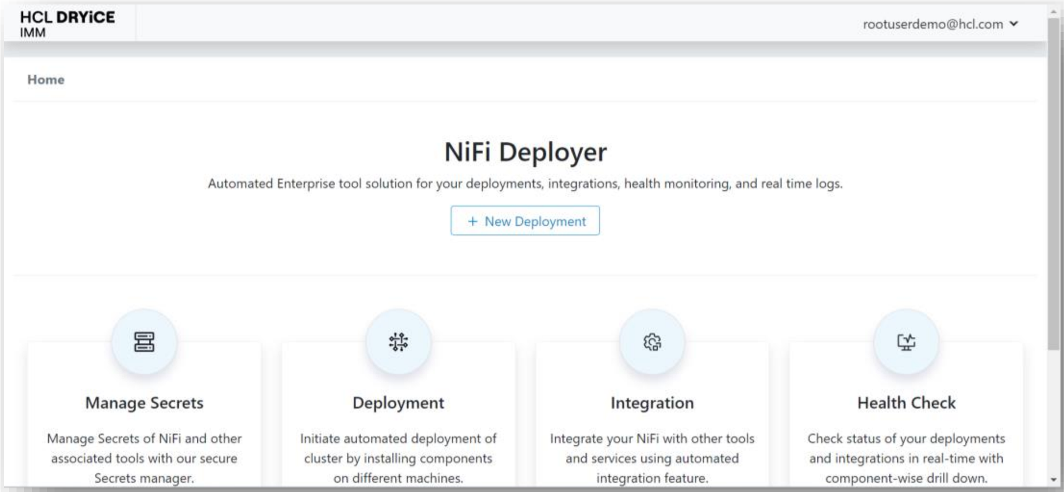
19. The Installation success message appears. It also prints the website URL.



```
Website URL - https://[redacted]/

IMM installtion has been completed successfully.
```

Figure 46 - Installation Success Message

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



NiFi Deployer

Automated Enterprise tool solution for your deployments, integrations, health monitoring, and real time logs.

Manage Secrets – Manage Secrets of NiFi and other associated tools with our secure Secrets manager.

Deployment – Initiate automated deployment of cluster by installing components on different machines.

Integration – Integrate your NiFi with other tools and services using automated integration feature.

Health Check – Check status of your deployments and integrations in real-time with component-wise drill down.

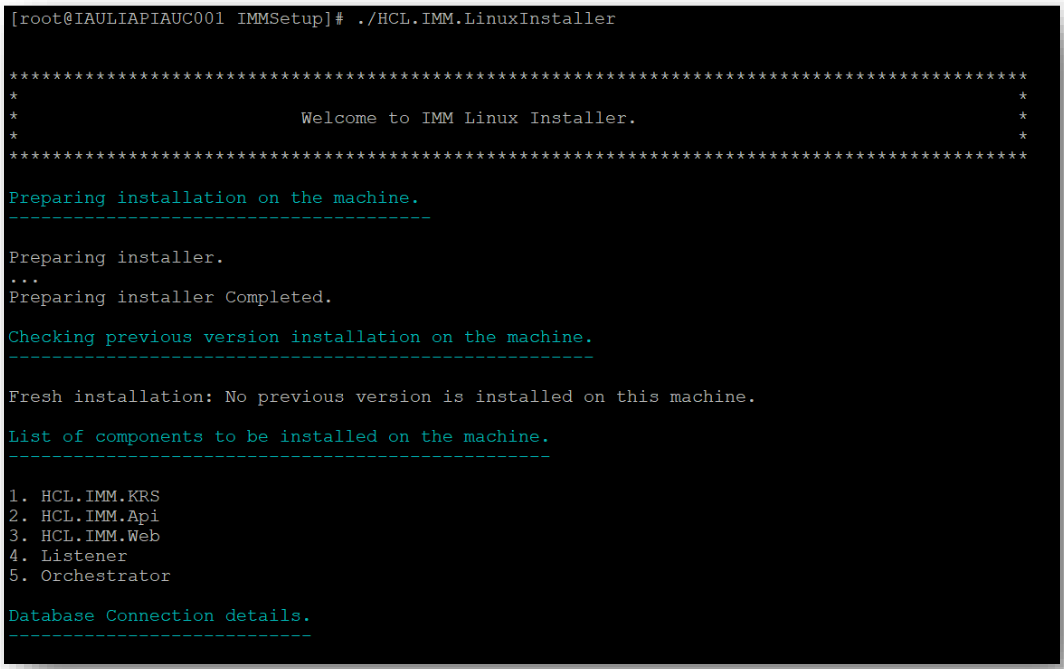
Figure 47 – IMM Application.

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:

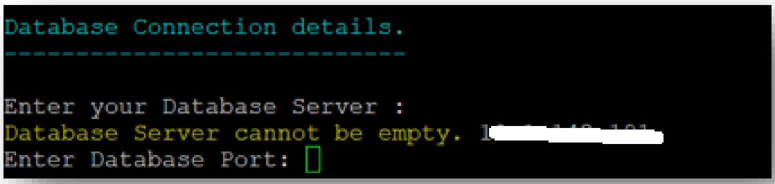


```
[root@IAULIAPIAUC001 IMMSetup]# ./HCL.IMM.LinuxInstaller

*********************************************************************************************
*                                                                                           *
*                           Welcome to IMM Linux Installer.                                 *
*                                                                                           *
*********************************************************************************************

Preparing installation on the machine.
---------------------------------------

Preparing installer.
...
Preparing installer Completed.

Checking previous version installation on the machine.
------------------------------------------------------

Fresh installation: No previous version is installed on this machine.

List of components to be installed on the machine.
--------------------------------------------------

1. HCL.IMM.KRS
2. HCL.IMM.Api
3. HCL.IMM.Web
4. Listener
5. Orchestrator

Database Connection details.
----------------------------
```

Figure 48 - 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 48 - 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.



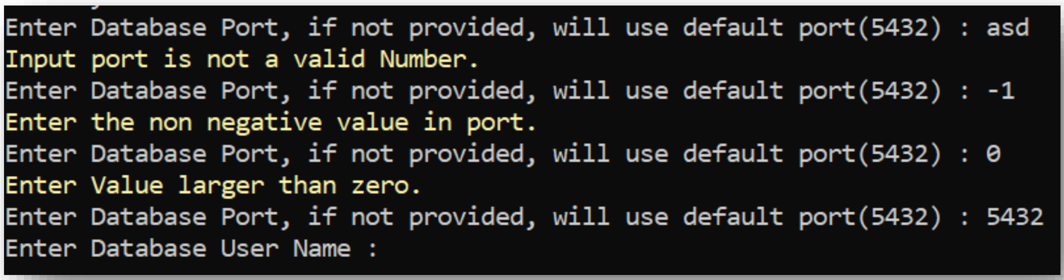
```
Database Connection details.
----------------------------

Enter your Database Server :
Database Server cannot be empty. 1[redacted]
Enter Database Port: 
```

Figure 49 – 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.

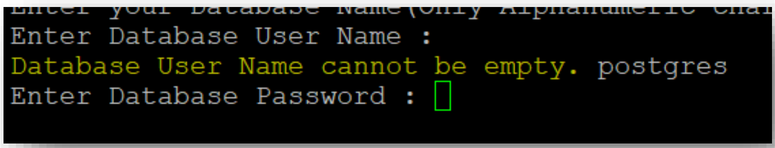


```
Enter Database Port, if not provided, will use default port(5432) : asd
Input port is not a valid Number.
Enter Database Port, if not provided, will use default port(5432) : -1
Enter the non negative value in port.
Enter Database Port, if not provided, will use default port(5432) : 0
Enter Value larger than zero.
Enter Database Port, if not provided, will use default port(5432) : 5432
Enter Database User Name :
```

Figure 50 – Database Connection Details, Database Port.

5. Input **Database Username**.

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



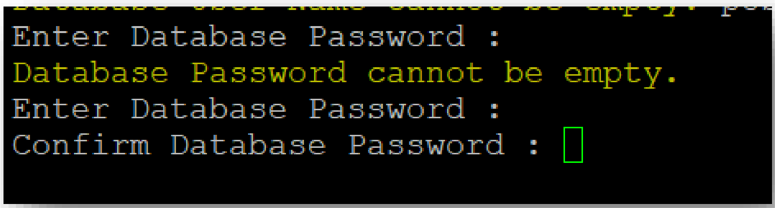
```
Enter Database User Name :
Database User Name cannot be empty. postgres
Enter Database Password : 
```

Figure 51 – Database Connection Details, Database Username.

6. Input **Database Password**.

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

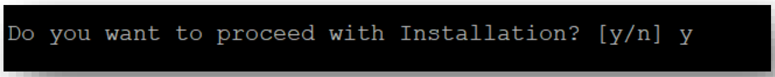
The input password is masked on screen for security purposes.



```
Enter Database Password :
Database Password cannot be empty.
Enter Database Password :
Confirm Database Password : 
```

Figure 52 – Database Connection Details, Database Password

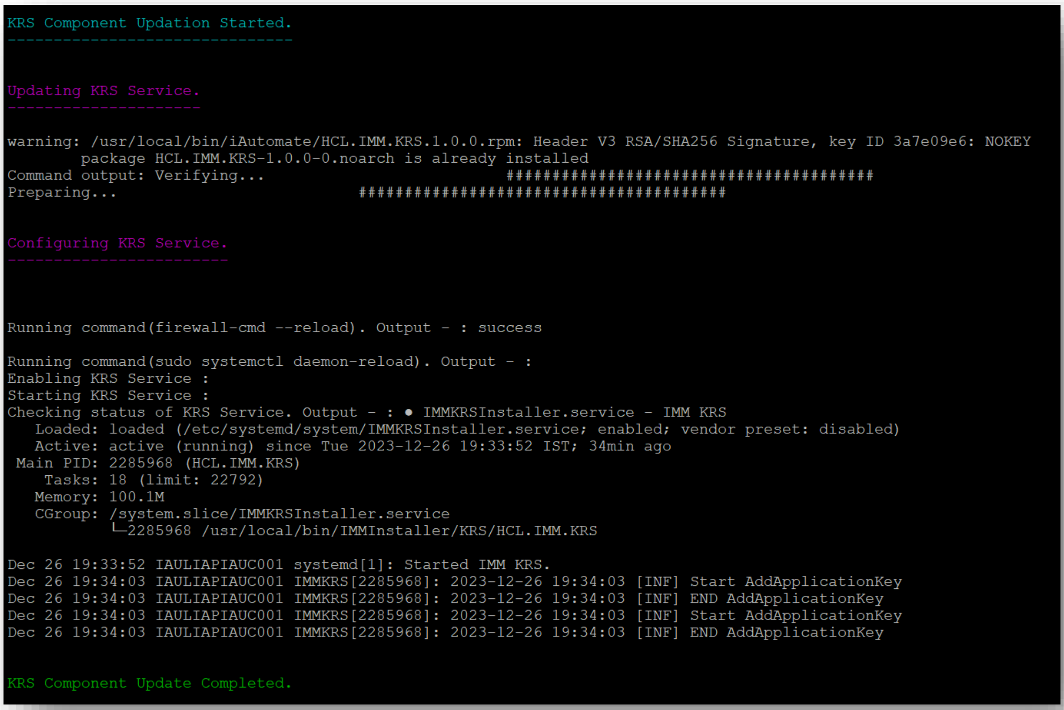
7. The installer requests a final confirmation from the user before proceeding with the installation.



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

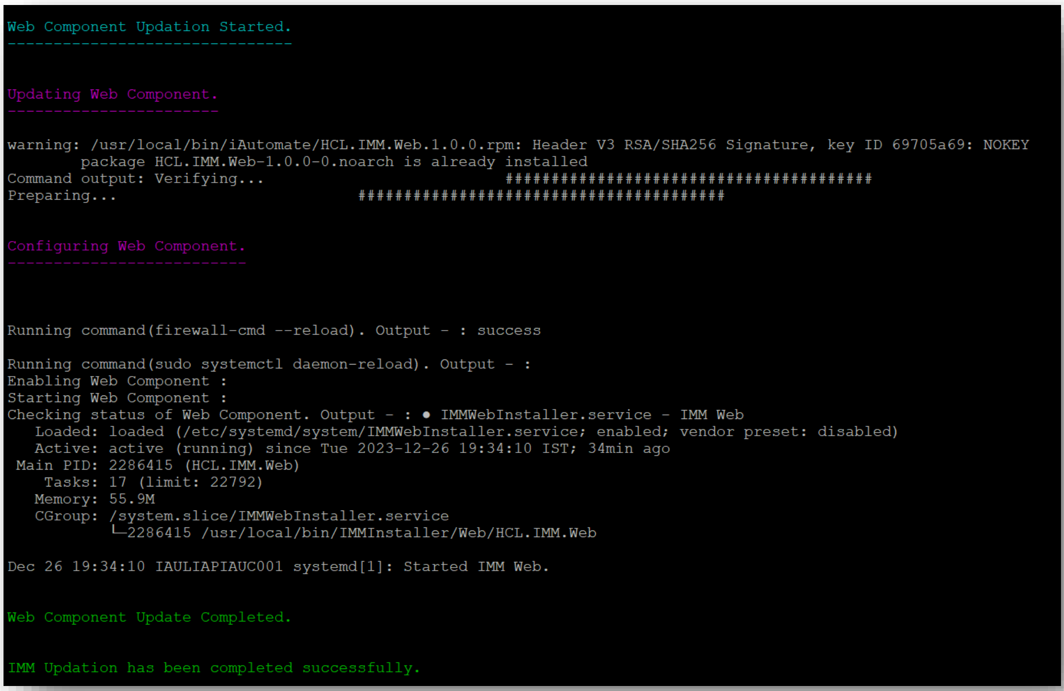
Figure 53 - Installation confirmation before proceeding further

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 54 – KRS Component Upgradation



```
Web Component Updation Started.
-------------------------------


Updating Web Component.
-----------------------

warning: /usr/local/bin/iAutomate/HCL.IMM.Web.1.0.0.rpm: Header V3 RSA/SHA256 Signature, key ID 69705a69: NOKEY
        package HCL.IMM.Web-1.0.0-0.noarch is already installed
Command output: Verifying...                         ########################################
Preparing...                          ########################################


Configuring Web Component.
--------------------------



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

Running command(sudo systemctl daemon-reload). Output - :
Enabling Web Component :
Starting Web Component :
Checking status of Web Component. Output - : ● IMMWebInstaller.service - IMM Web
   Loaded: loaded (/etc/systemd/system/IMMWebInstaller.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2023-12-26 19:34:10 IST; 34min ago
 Main PID: 2286415 (HCL.IMM.Web)
    Tasks: 17 (limit: 22792)
   Memory: 55.9M
   CGroup: /system.slice/IMMWebInstaller.service
           └─2286415 /usr/local/bin/IMMInstaller/Web/HCL.IMM.Web

Dec 26 19:34:10 IAULIAPIAUC001 systemd[1]: Started IMM Web.


Web Component Update Completed.


IMM Updation has been completed successfully.
```

Figure 55 – Web Component Upgradation

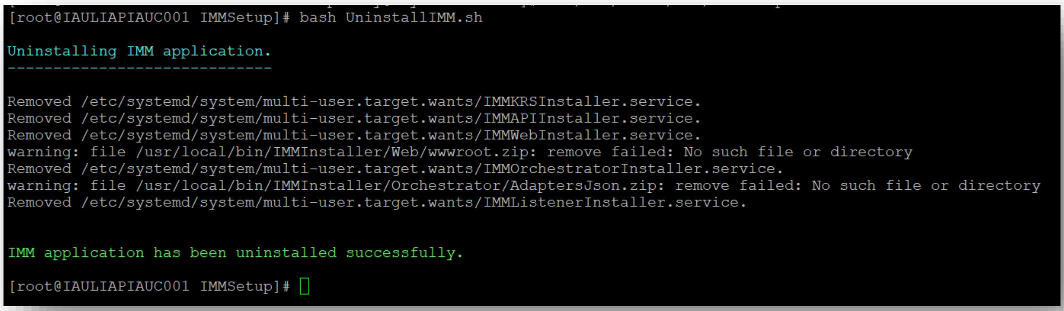
2.8.4 IMM Uninstallation

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
```



```
[root@IAULIAPIAUC001 IMMSetup]# bash UninstallIMM.sh

Uninstalling IMM application.
-----------------------------

Removed /etc/systemd/system/multi-user.target.wants/IMMKRSInstaller.service.
Removed /etc/systemd/system/multi-user.target.wants/IMMAPIInstaller.service.
Removed /etc/systemd/system/multi-user.target.wants/IMMWebInstaller.service.
warning: file /usr/local/bin/IMMInstaller/Web/wwwroot.zip: remove failed: No such file or directory
Removed /etc/systemd/system/multi-user.target.wants/IMMOrchestratorInstaller.service.
warning: file /usr/local/bin/IMMInstaller/Orchestrator/AdaptersJson.zip: remove failed: No such file or directory
Removed /etc/systemd/system/multi-user.target.wants/IMMListenerInstaller.service.


IMM application has been uninstalled successfully.

[root@IAULIAPIAUC001 IMMSetup]#
```

Figure 56 – IMM Components Uninstallation.

4. The IMM application is uninstalled successfully.

HCLSoftware

hcltechsw.com