

HCLSoftware

HCL IntelliOps Event Management

Collector Installation Guide

Version 1.1



The data contained in this document shall not be duplicated, used, or disclosed in whole or in part for any purpose. If a contract is awarded to chosen parties because of or in connection with the submission of this data, the client or prospective client shall have the right to duplicate, use, or disclose this data to the extent provided in the contract. This restriction does not limit the client's or prospective client's right to use the information contained in the data if it is obtained from another source without restriction. The data subject to this restriction is contained in all marked sheets.

HCL has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the HCL website at www.hcltechsw.com.

Copyright © 2025 HCL Technologies Limited.

Contents

1	Preface	8
1.1	Intended Audience	8
1.2	About this Guide	8
1.3	Related Documents	8
1.4	Conventions	8
2	IEM Collector	9
2.1	Overview for NiFi	9
2.1.1	NiFi Architecture	9
2.2	Prerequisites	10
2.2.1	System Requirements For NiFi	11
2.2.2	Supported OS for NIFI	11
2.2.3	Supported Web Browsers	11
2.2.4	Hardware Sizing Recommendation	11
2.2.5	Port Requirement for NiFi	12
2.4	NiFi Installation and Setup	12
2.4.1	Pre-requisites of NiFi Installation	12
2.4.2	Installation of NiFi Cluster	15
2.5	Overview of IMM	25
2.6	Prerequisite for IMM	25
2.6.1	Supported OS for IMM	25
2.6.2	Supported Web Browsers	25
2.6.3	Hardware Sizing Recommendation	26
2.6.4	Port Requirement for IMM	26
2.7	IMM Installation and Setup	26
2.7.1	IMM Components	26
2.7.2	IMM Installation	27
2.7.3	IMM Upgradation	35
2.7.4	IMM Uninstallation	38

Table of Figures

Figure 1 - Content and Provenance repository	9
Figure 2 - Repositories.....	10
Figure 3 – Log -in	25
Figure 4 – Login to Server.....	27
Figure 5 – Elevate the user.....	27
Figure 6 – Navigate to Installer path.	27
Figure 7 – Provide access to file.	27
Figure 8 - Run the Installer	28
Figure 9 - IMM Installer	28
Figure 10 – Database Connection Details, Database Server.	29
Figure 11 – Database Connection Details, Database Port.	29
Figure 12 – Database Connection Details, Database Username.	29
Figure 13 – Database Connection Details, Database password.	29
Figure 14 – Database Connection Validation – Connection Failed.....	30
Figure 15 – Database Connection Validation – Connection Succes.	30
Figure 16 – Root User details – Root Username	30
Figure 17 – Root User details – First Name	31
Figure 18 – Root User details – Last Name.....	31
Figure 19 – Root User details – Root User Email.....	31
Figure 20 – Root User details – Root User Password.	32
Figure 21 - Root User details – Confirm Root User Password.	32
Figure 22 – Input port for KRS, API, Web, and Orchestrator applications	32
Figure 23 – Installation confirmation before proceeding further.	32
Figure 24 – Installation Database Check and Other Database Tasks.....	33
Figure 25 – KRS Component Installation	33
Figure 26 – API Component Installation.....	34
Figure 27 – Web Component installation.....	34
Figure 28 - Installation Success Message	34
Figure 29 – IMM Application.	35

Figure 30 - IMM Installer – Upgrade35

Figure 31 – Database Connection Details, Database Server36

Figure 32 – Database Connection Details, Database Port.....36

Figure 33 – Database Connection Details, Database Username.36

Figure 34 – Database Connection Details, Database Password36

Figure 35 - Installation confirmation before proceeding further.37

Figure 36 – KRS Component Upgradation37

Figure 37 – Web Component Upgradation38

Figure 38 – IMM Components Uninstallation.38

List of Tables

Table 1 - Conventions8

Table 2 - zookeeper.properties18

Table 3 - Types of Servers.....26

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

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.

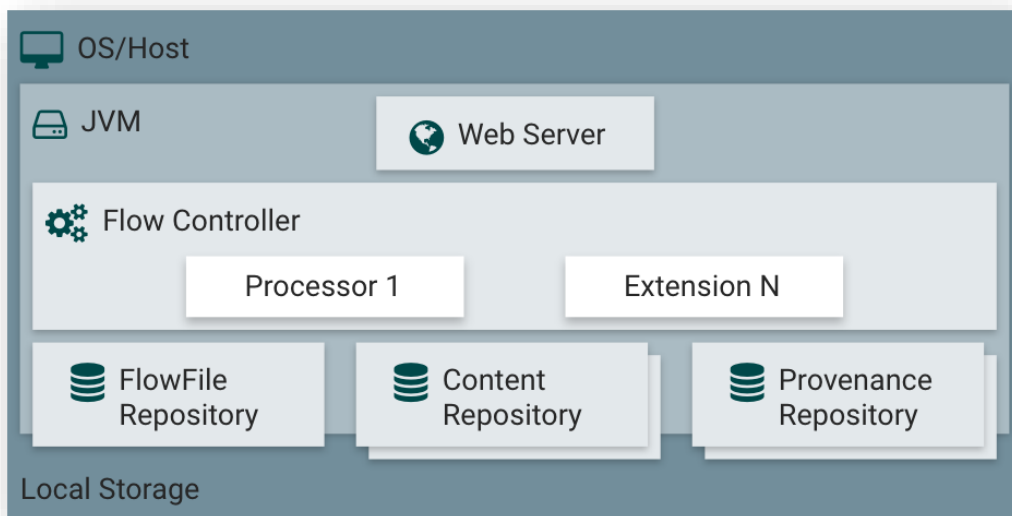


Figure 1 - Content and Provenance repository

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 construct is pluggable with the default implementation being to use one or more physical disk volumes. Within each location event data is indexed and searchable.

NiFi is also able to operate within a cluster.

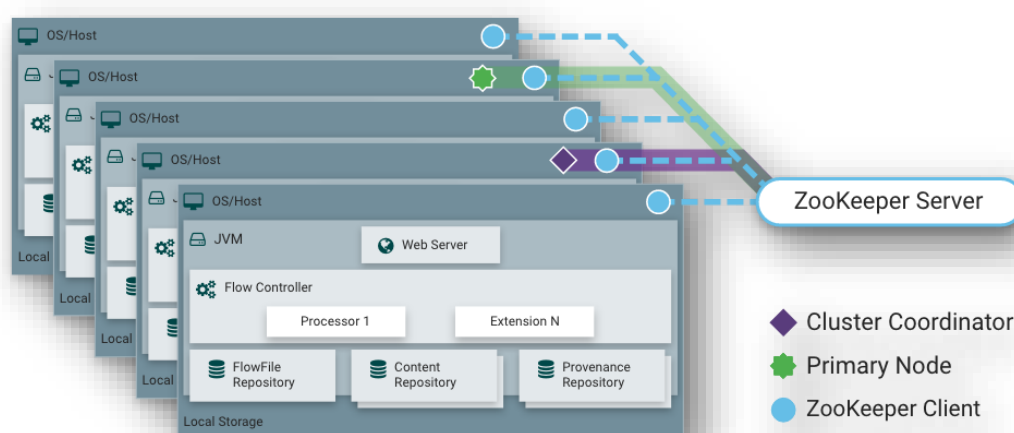


Figure 2 - Repositories

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)

Required Sustained Throughput	Minimum Hardware Requirement
85 events per second	<ul style="list-style-type: none">• 3nodes• 4 or more cores per node• 6 or more disks per node (SSD or spinning)• 8 GB memory per node

	<ul style="list-style-type: none"> • 1 GB bonded NICs
114 events per second	<ul style="list-style-type: none"> • 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 is (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-W-01-P002 ~]# useradd nifiadmin
[root@S-W-01-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 tell 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
```

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

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

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

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

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

- Switch to “nifiadmin” user to complete the installation steps.
- Run the below command (on all nodes) for modifying a user account:

```
su - nifiadmin
```

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

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

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

```
[nifiadmin@S.....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.....P002 ~]$
```

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

```
java -version
```

```
[nifiadmin@S... 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... P002 ~]$
[nifiadmin@S... P002 ~]$
```

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

```
[nifiadmin@S... P002 ~]$ cat /etc/hosts
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
#Address <SPACE> FQDN <SPACE> Hostname
#...
10.1.152.204 ... P002.d... labs.com ... P002
```

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... P002 ~]$ ping S... P001.d... labs.com
PING S... P001.dryicelabs.com (10.1.1...7) 56(84) bytes of data.
64 bytes from S... P001.d... labs.com (10.1.1...7): icmp_seq=1 ttl=64 time=0.194 ms
64 bytes from S... P001.d... labs.com (10.1.1...7): icmp_seq=2 ttl=64 time=0.163 ms
64 bytes from S... P001.d... labs.com (10.1.1...7): icmp_seq=3 ttl=64 time=0.216 ms
64 bytes from S... P001.d... labs.com (10.1.1...7): icmp_seq=4 ttl=64 time=0.221 ms
^C
--- S... P001.d... 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... P002 ~]$ ping S... P003.d... labs.com
PING S... P003.d... labs.com (10.1.1...9) 56(84) bytes of data.
64 bytes from S... P003.d... labs.com (10.1.1...9): icmp_seq=1 ttl=64 time=0.199 ms
64 bytes from S... P003.d... labs.com (10.1.1...9): icmp_seq=2 ttl=64 time=0.170 ms
^C
--- S... P003.d... 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 CMDDB data:

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

```
[nifiadmin@P002 ~]$ mkdir /home/nifiadmin/script/
[nifiadmin@P002 ~]$
[nifiadmin@P002 ~]$
[nifiadmin@P002 ~]$ mkdir /home/nifiadmin/script/cmdb-ci/
[nifiadmin@P002 ~]$
[nifiadmin@P002 ~]$
[nifiadmin@P002 ~]$ touch /home/nifiadmin/script/cmdb-ci/cmdbci-repo
[nifiadmin@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@ ~]$ 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  0:00:02  0:00:02  0:00:00  5055k
  9 1409M    9 139M    0     0  5135k      0  0:00:02  0:00:02  0:00:00  5135k
10 1409M   10 147M    0     0  5233k      0  0:00:02  0:00:02  0:00:00  5233k
10 1409M   10 154M    0     0  5304k      0  0:00:02  0:00:02  0:00:00  5304k
11 1409M   11 161M    0     0  5352k      0  0:00:02  0:00:02  0:00:00  5352k
11 1409M   11 168M    0     0  5412k      0  0:00:02  0:00:02  0:00:00  5412k
100 1409M  100 1409M    0     0  6152k      0  0:00:02  0:00:02  0:00:00  6152k
[nifiadmin@ ~]$
```

```
[nifiadmin@ ~]$ 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@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@P001 ~]$
```

```
[nifiadmin@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@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@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@P001 ~]$ mkdir -p sslcerts
[nifiadmin@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@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 standalone -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 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.standalone.TlsToolkitStandalone - Generated new CA certificate ../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
│   ├── P002.d labs.com
│   │   ├── keystore.jks
│   │   ├── nifi.properties
│   │   └── truststore.jks
│   └── 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@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 .....P001.d ..... labs.com
drwx----- 2 nifiadmin nifiadmin 71 Sep 27 16:01 .....P002.d ..... labs.com
drwx----- 2 nifiadmin nifiadmin 71 Sep 27 16:01 .....P003.d ..... labs.com
[nifiadmin@P001 sslcerts]$ cp .....P001.d ..... labs.com/* /home/nifiadmin/nifi-1.23.2/conf/
[nifiadmin@P001 sslcerts]$ scp .....P002.d ..... labs.com/* .....P002.d ..... labs.com:/home/nifiadmin/nifi-1.23.2/conf/
nifiadmin@P002.d ..... 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@P001 sslcerts]$ scp .....P003.d ..... labs.com/* .....P003.d ..... labs.com:/home/nifiadmin/nifi-1.23.2/conf/
nifiadmin@P003.d ..... 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@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@P002 ~]$ cd nifi-1.23.2/conf/
[nifiadmin@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-rw-r-- 1 nifiadmin nifiadmin 3142 Sep 27 17:49 keystore.jks
-rw-rw-r-- 1 nifiadmin nifiadmin 17445 Sep 27 17:49 nifi.properties
-rw-rw-r-- 1 nifiadmin nifiadmin 935 Sep 27 17:49 truststore.jks
[nifiadmin@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 2 - 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@P001 ~]$ cd /home/nifiadmin/nifi-1.23.2/
[nifiadmin@P001 nifi-1.23.2]$ mkdir -p ./state/zookeeper
[nifiadmin@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@P001 nifi-1.23.2]$ cd state/
[nifiadmin@P001 state]$ ls -lrt
total 0
drwxrwxr-x. 2 nifiadmin nifiadmin 6 Sep 28 14:14 zookeeper
[nifiadmin@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@... P001 state]$ touch /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@... P001 state]$ vim /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@... P001 state]$ cat /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
1
```

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

```
[nifiadmin@... P003 ~]$ touch /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@... P003 ~]$ vim /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
[nifiadmin@... P003 ~]$ cat /home/nifiadmin/nifi-1.23.2/state/zookeeper/myid
3
[nifiadmin@... P003 ~]$
```

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 below configuration 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.c
om: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.co
m: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@.....0003 ~]$ mkdir /home/nifiadmin/sslcerts/
[nifiadmin@.....003 ~]$ mkdir /home/nifiadmin/sslcerts/nifisummaryapi
[nifiadmin@l.....0003 ~]$ exit
logout
Connection to .....05 closed.
[nifiadmin@F.....002 ~]$ ssh .....
nifiadmin@l.....'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: ..... 17:38:00 2023 from 172.16.0.89
[nifiadmin@.....0002 ~]$ mkdir /home/nifiadmin/sslcerts/
[nifiadmin@.....002 ~]$ mkdir /home/nifiadmin/sslcerts/nifisummaryapi
[nifiadmin@l.....0002 ~]$ exit
logout
Connection to l..... 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@nifi ~]$ cd /opt/nifi-standalone-1.9.0/bin/tls-toolkit.sh standalone -s "nifissummaryapi" --subjectAlternativeNames "*"
[nifiadmin@nifi ~]$ ./tls-toolkit.sh standalone -s "nifissummaryapi" --subjectAlternativeNames "*" --key "key.pem"
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - 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.ssl.TrustBuilder - Verifying the certificate signautre for CN=localhost,O=HIFI
[main] INFO org.apache.nifi.toolkit.tls.ssl.TrustBuilder - Attempting to verify certificate CN=localhost,O=HIFI signiture with CN=localhost,O=HIFI
[main] INFO org.apache.nifi.toolkit.tls.ssl.TrustBuilder - Certificate was signed by CN=localhost,O=HIFI
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Using existing CA certificate ../sslcerts/nifi-cert.pem and key ../sslcerts/nifi-key.key
[warn] WARN org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Hostname count does not match given alternate name count. Verify names in resulting certificates
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Writing new ssl configuration to ../sslcerts/nifissummaryapi
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - Successfully generated TLS configuration for nifissummaryapi in ../sslcerts/nifissummaryapi
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - No clientCerts specified, not generating any client certificates.
[main] INFO org.apache.nifi.toolkit.tls.standalone.TlsToolkitStandalone - tls-toolkit standalone completed successfully
[nifiadmin@nifi ~]$ cd /opt/nifi-standalone-1.9.0/bin/tls-toolkit.sh standalone ls -lrt
total 8
-rw-r--r-- 1 nifiadmin nifiadmin 1296 Dec 20 15:43 nifi-cert.pem
-rw-r--r-- 1 nifiadmin nifiadmin 1878 Dec 20 15:43 nifi-key.key
-rwxr-xr-x 1 nifiadmin nifiadmin 71 Dec 20 15:43
-rwxr-xr-x 1 nifiadmin nifiadmin 71 Dec 20 15:43
-rwxr-xr-x 1 nifiadmin nifiadmin 71 Dec 20 15:43
-rwxr-xr-x 1 nifiadmin nifiadmin 71 Dec 20 15:43
-rwxr-xr-x 1 nifiadmin nifiadmin 71 Dec 20 15:43
[nifiadmin@nifi ~]$ cd /opt/nifi-standalone-1.9.0/bin/tls-toolkit.sh standalone ls -lrt
total 28
-rw-r--r-- 1 nifiadmin nifiadmin 936 Dec 20 17:46 truststore.jks
-rw-r--r-- 1 nifiadmin nifiadmin 3203 Dec 20 17:46 keystore.jks
-rwxr-xr-x 1 nifiadmin nifiadmin 17400 Dec 20 17:46 nifi.properties
[nifiadmin@nifi ~]$ cd /opt/nifi-standalone-1.9.0/bin/tls-toolkit.sh standalone ls -lrt
total 16
-rw-r--r-- 1 nifiadmin nifiadmin 101 Dec 20 17:46 nifi.properties
[nifiadmin@nifi ~]$ cd /opt/nifi-standalone-1.9.0/bin/tls-toolkit.sh standalone ls -lrt
total 16
-rw-r--r-- 1 nifiadmin nifiadmin 101 Dec 20 17:46 nifi.properties
```

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:

- `rwrxwxr-x 1 nifiadmin nifiadmin 3876 Dec 28 19:28 trapv3fornifi.py`
- `rwrxwxr-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`
- `rwrxwxr-x 1 nifiadmin nifiadmin 1517 Dec 28 19:32 context.json`
- `rwrx-r-x 1 nifiadmin nifiadmin 3721 Dec 28 19:33 publish-accesstoken.py`

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

```
[nifiadmin@17 script]$ ls -ltr
total 20
-rwxrwxr-x 1 nifiadmin nifiadmin 3876 Dec 28 19:28 trapv3fornifi.py
drwxrwxr-x 2 nifiadmin nifiadmin 91 Dec 28 19:28 cmd-db-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 node.

```
[nifiadmin@nifi17 lib]$ ls -lrt
total 1060
-rwxrwxr-x 1 nifiadmin nifiadmin 1081604 Dec 28 19:37 postgresql-42.6.0.jar
[nifiadmin@nifi17 lib]$
[nifiadmin@nifi17 lib]$ pwd
/home/nifiadmin/nifi-1.17.0/lib
[nifiadmin@nifi17 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@nifi17 ~]$ 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 running 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@nifi17 ~]$ sudo sh nifi-1.23.2/bin/nifi.sh install
Service nifi 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@S...P002 ~]$ 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...P002 ~]$ 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:

- 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...P001 ~]$ scp -r /home/nifiadmin/nifi-toolkit-1.23.2/ S'...P003:/home/nifiadmin
nifiadmin@S...P001 ~$ scp -r /home/nifiadmin/nifi-toolkit-1.23.2/ S'...P003:/home/nifiadmin
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
```

- 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...P001 ~]$ vim /home/nifiadmin/nifi-1.23.2/conf/nifi.properties
[nifiadmin@S...P001 ~]$ /home/nifiadmin/nifi-toolkit-1.23.2/bin/encrypt-config.sh -b /home/nifiadmin/nifi-1.23.2/conf/bootstrap.conf -k 0123456789ABCDEFFEDCBA98765432100123456789ABCDEFFEDCBA9876543210 -n /home/nifiadmin/nifi-1.23.2/conf/nifi.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 are 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.privateLookupIn. Will attempt to fallback to using the package-private constructor.
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Protected [nifi.security.keyPasswd] using [aes/gcm] -> kvf89JrGortaae8Xj[REDACTED]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Updated protection key [nifi.security.keyPasswd] using [aes/gcm] -> KJxaBx[FVUmh+H1Z][REDACTED]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Protected [nifi.security.keystorePasswd] using [aes/gcm] -> h0t0Bg8vCem0j+1[Ky2De7Un6S[REDACTED]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Updated protection key [nifi.security.keystorePasswd] using [aes/gcm] -> MPj1f4Jb27[REDACTED]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Protected [nifi.sensitive.props.key] using [aes/gcm] -> 11RUSw/bE+cJTFVN][REDACTED]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Updated protection key [nifi.sensitive.props.key] using [aes/gcm] -> 70XXRho3fk[REDACTED]
[main] INFO org.apache.nifi.properties.ConfigEncryptionTool - Final result: 208 keys including 4 protected keys
[nifiadmin@S...P001 ~]$
```

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||IN7u5OXU1E7z+cy2q8S1euROupbmL+kBZ8a6Tio3us2dwA==
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.

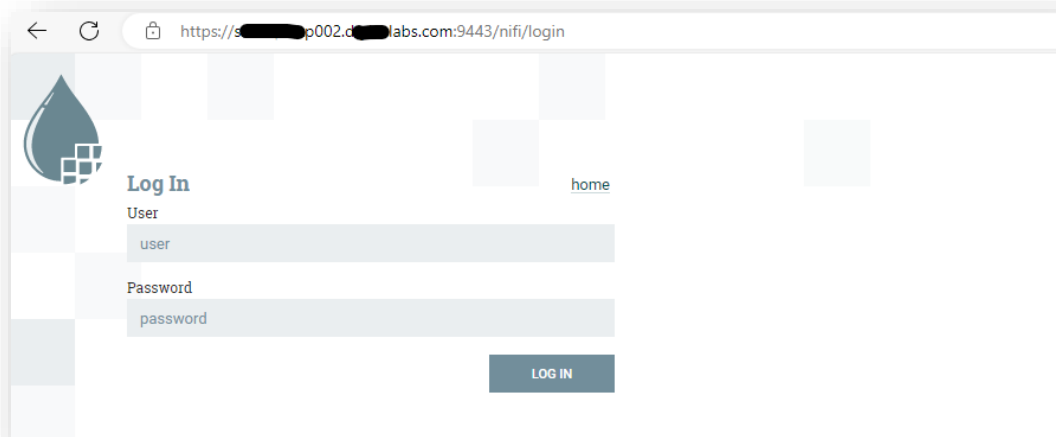


Figure 3 – Log -in

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 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

2.6.3 Hardware Sizing Recommendation

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

2.6.4 Port Requirement for IMM

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

2.7 IMM Installation and Setup

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

2.7.1 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 3 - 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

2.7.2 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.

2.7.2.1 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:

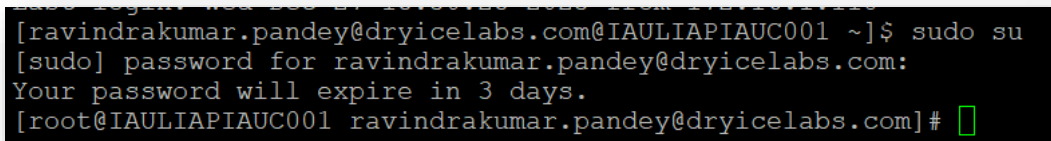
1. Open putty and Login with valid credentials to the targeted server where you want to install IMM Application.



```
root@IAULIAPIAUC001:~/home/ravindrakumar.pandey@dryicelabs.com
login as: ravindrakumar.pandey@dryicelabs.com
ravindrakumar.pandey@dryicelabs.com'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 from [REDACTED]
```

Figure 4 – Login to Server.

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



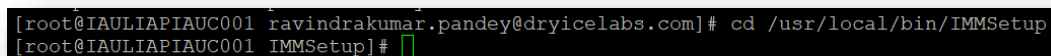
```
[ravindrakumar.pandey@dryicelabs.com@IAULIAPIAUC001 ~]$ sudo su
[sudo] password for ravindrakumar.pandey@dryicelabs.com:
Your password will expire in 3 days.
[root@IAULIAPIAUC001 ravindrakumar.pandey@dryicelabs.com]#
```

Figure 5 – 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



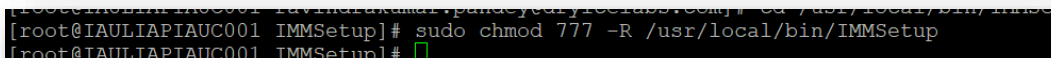
```
[root@IAULIAPIAUC001 ravindrakumar.pandey@dryicelabs.com]# cd /usr/local/bin/IMMSetup
[root@IAULIAPIAUC001 IMMSetup]#
```

Figure 6 – 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 7 – 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 8 - Run the Installer

2.7.2.2 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 3 - 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 9 - 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 9 - 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. 192.168.1.101
Enter Database Port: 

```

Figure 10 – 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 11 – Database Connection Details, Database Port.

5. Input the **Database User Name**.

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 12 – 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. postgres
Enter Database Password : 
Database Password cannot be empty.
Enter Database Password : 
Confirm Database Password : 

```

Figure 13 – 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 proceed 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 14 – 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 : 192.168.1.101
Enter Database Port: 5432
Enter your Database Name(Only Alphanumeric characters and Underscore is allowed) : imddblinuxinstaller
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 15 – Database Connection Validation – Connection Success.

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

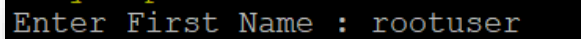
```

Figure 16 – 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.



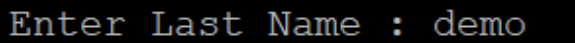
```
Enter First Name : rootuser
```

Figure 17 – 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.

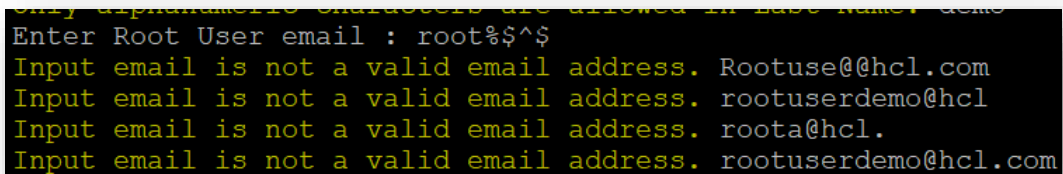


```
Enter Last Name : demo
```

Figure 18 – 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 19 – 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 20 – 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 21 - 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 22 – 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 23 – 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.
-----

Fresh 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 24 – Installation Database Check and Other Database Tasks

The components installation starts.

```
KRS Component Installation 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 Installation Completed.
```

Figure 25 – KRS Component Installation

```

API Component Installation 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 IAULIAPITAU001 systemd[1]: Started IMM API.

API Component Installation Completed.

```

Figure 26 – API Component Installation

```

Web Component Installation 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 IAULIAPITAU001 systemd[1]: Started IMM Web.

Web Component Installation Completed.

```

Figure 27 – Web Component installation.

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

```

Website URL - https://192.168.55.102:4000/
IMM installtion has been completed successfully.

```

Figure 28 - Installation Success Message

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

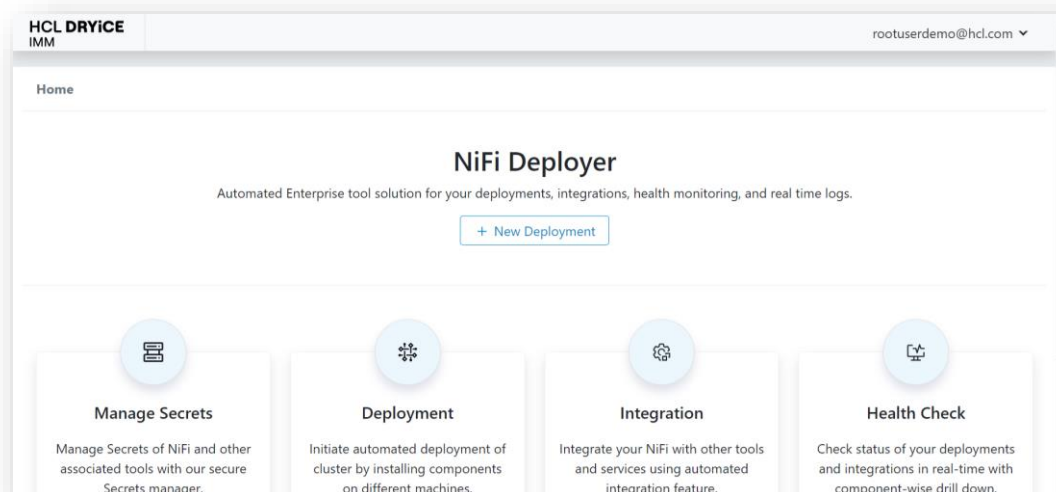


Figure 29 – IMM Application.

2.7.3 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 30 - 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 30 - 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. 192.168.1.101
Enter Database Port: 

```

Figure 31 – 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 32 – Database Connection Details, Database Port.

5. Input **Database Username**.

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 33 – 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.

```

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

```

Figure 34 – 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 35 - 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 36 – 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 37 – Web Component Upgradation

2.7.4 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 38 – IMM Components Uninstallation.

4. The IMM application is uninstalled successfully.

HCLSoftware

hcltechsw.com