Resonate RFID Reader Management

Version 2.0

Software Installation Guide

ZEBRA

MN-005495-01EN Rev B

Copyright

2025/09/05

Terms of Use

Proprietary Statement

This manual contains proprietary information of Zebra Technologies Corporation and its subsidiaries (“Zebra Technologies”). It is intended solely for the information and use of parties operating and maintaining the equipment described herein. Such proprietary information may not be used, reproduced, or disclosed to any other parties for any other purpose without the express, written permission of Zebra Technologies.

Product Improvements

Continuous improvement of products is a policy of Zebra Technologies. All specifications and designs are subject to change without notice.

Liability Disclaimer

Zebra Technologies takes steps to ensure that its published Engineering specifications and manuals are correct; however, errors do occur. Zebra Technologies reserves the right to correct any such errors and disclaims liability resulting therefrom.

Limitation of Liability

In no event shall Zebra Technologies or anyone else involved in the creation, production, or delivery of the accompanying product (including hardware and software) be liable for any damages whatsoever (including, without limitation, consequential damages including loss of business profits, business interruption, or loss of business information) arising out of the use of, the results of use of, or inability to use such product, even if Zebra Technologies has been advised of the possibility of such damages. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

Contents

About this Guide

This guide provides information about installing Resonate RFID Reader Management.

Icon Conventions

The documentation set is designed to give the reader more visual clues. The following visual indicators are used throughout the documentation set.

NOTE: The text here indicates information that is supplemental for the user to know and that is not required to complete a task.

IMPORTANT: The text here indicates information that is important for the user to know.

CAUTION: If the precaution is not heeded, the user could receive a minor or moderate injury.

WARNING: If danger is not avoided, the user CAN be seriously injured or killed.

DANGER: If danger is not avoided, the user WILL be seriously injured or killed.

Notational Conventions

The following notational conventions make the content of this document easy to navigate.

- **Bold** text is used to highlight the following:
  - Dialog box, window, and screen names
  - Dropdown list and list box names
  - Checkbox and radio button names
  - Icons on a screen
  - Key names on a keypad
  - Button names on a screen

- Bullets (•) indicate:
  - Action items
  - List of alternatives
  - Lists of required steps that are not necessarily sequential
- Sequential lists (for example, those that describe step-by-step procedures) appear as numbered lists.

Introduction

This section provides an overview of Resonate RFID Reader Management, its installation options, its network architecture, and how the software integrates with your asset tracking solution.

Overview

Resonate RFID Reader Management is a powerful, scalable platform to deploy, monitor, manage, and configure the RFID readers in your asset tracking solution.

Resonate RFID Reader Management can run either distributed across multiple Linux machines (multi-node server) or on a single Linux machine (single-node server). Multi-node is ideal for asset-tracking solutions requiring high availability and minimum downtime, whereas single-node is suitable for smaller asset-tracking solutions, customer tests, and demos without these requirements.

You must install Resonate RFID Reader Management according to the required deployment mode: single-node or multi-node; this cannot be changed after installation. Multi-node mode requires at least 3 nodes initially; you can add more nodes after installation.

To achieve maximum scalability, Resonate RFID Reader Management is deployed and managed using MicroK8s, a lightweight Kubernetes distribution. MicroK8s is included and automatically set up during installation.

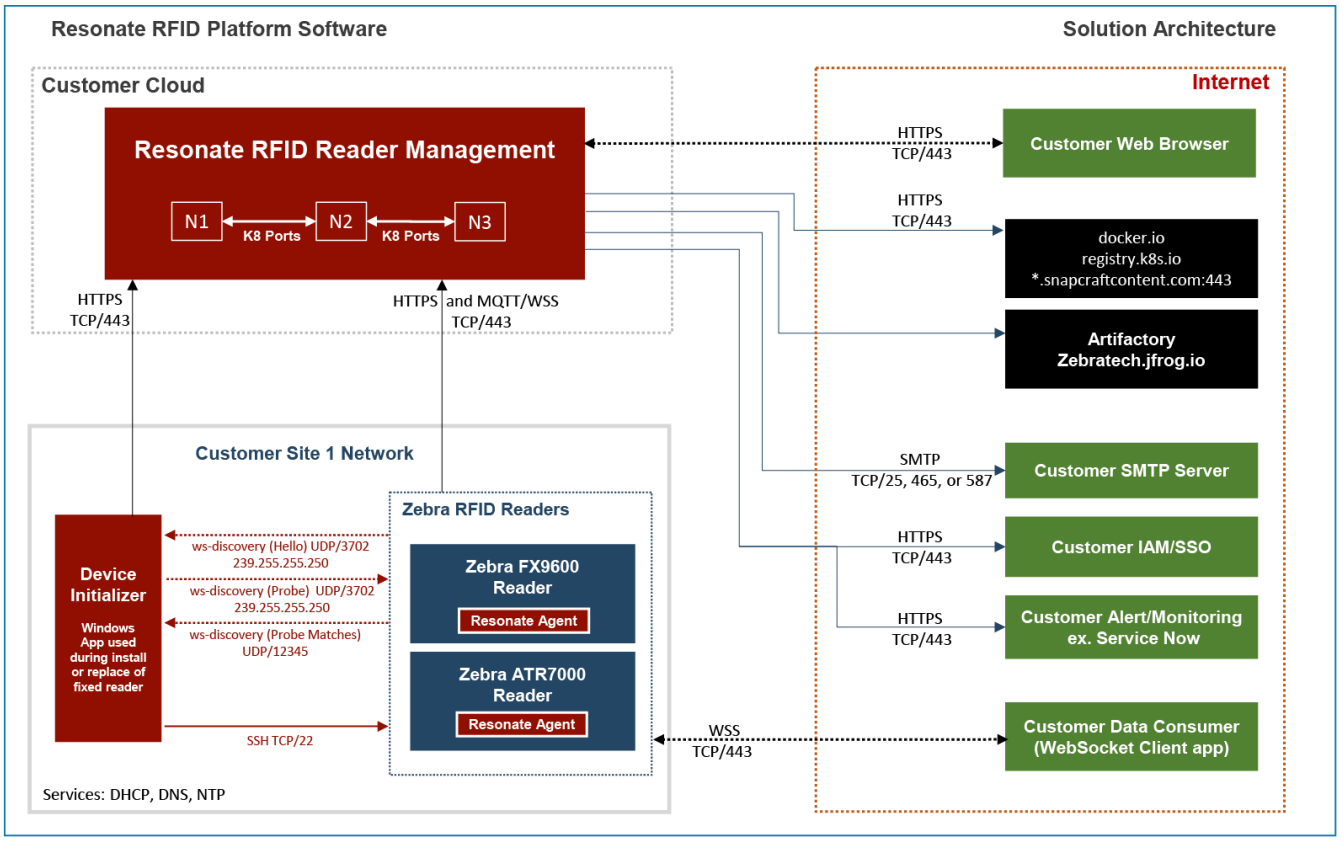
To manage user authentication and authorization, Resonate RFID Reader Management uses Keycloak. This setup allows Resonate to offer seamless single sign-on and role-based access control, ensuring a secure and user-friendly experience. The Keycloak service is included and automatically set up during installation.

For installation, Resonate RFID Reader Management offers both an online and an offline installer. The online installer is lightweight, using an internet connection to download Resonate containers and third-party prerequisites as needed. The offline installer runs without internet access and is useful if you need to install the software in a protected environment behind a corporate firewall.

Network Architecture and Integration of Resonate RFID Reader Management

The following illustrates the Resonate RFID Reader Management network architecture and how the software integrates with your asset-tracking solution.

Figure 1 Resonate RFID Reader Management Solution Architecture



Integrating Into Your Asset-Tracking Solution

The basic steps for integrating Resonate RFID Reader Management into your asset-tracking solution are outlined below. This guide deals with the first step. Refer to the user guide for information on the remaining steps.

In these steps, Resonate refers to Resonate RFID Reader Management, and your application refers to your application or partner's application that processes RFID reads.

1. Install Resonate on one or more Linux machines (nodes).

   When you run the online Resonate installer, it downloads Resonate containers from Zebra's Artifactory repository and downloads third-party open-source prerequisites as needed.

2. From a browser, log in to Resonate to configure it and eventually to manage your RFID readers.
3. If required, configure Resonate to use your Single Sign-On (SSO) provider (for example, IAM/SSO) for authentication.
4. Configure Resonate to send email alerts to administrator users.

5. Configure Resonate to send alert webhooks (https POST messages) to your IT case management solution.[1]
6. Run the Resonate Device Initializer on a Windows machine to initialize the readers that Resonate needs to manage, monitor, and configure; this also installs the Resonate Agent on the readers. For automatic device discovery, run the utility onsite and on the same subnet as your RFID readers; for an explicitly specified list of readers, you can run the utility from any Windows machine provided the network access is open[2]. The Resonate Device Initializer is a single executable without an installer and with no outside dependencies.
7. Connect business applications that consume RFID tag read data directly to each reader to subscribe to its data streams.

Kubernetes Environment

Resonate RFID Reader Management is deployed and managed using MicroK8s, a lightweight Kubernetes distribution. MicroK8s is responsible for deploying, scaling, and operating Resonate containers. These containers adhere to Docker standard formatting, which packages the application code and dependencies into isolated, portable units. MicroK8s allows for efficient scaling to meet varying demand, adjusting the number of Resonate containers running at any given time, on one or multiple nodes, depending on the installation.

NOTE: Resonate RFID Reader Management comes bundled with MicroK8s and is configured to use MicroK8s as required for its operation, testing, and support. References to Kubernetes in the Resonate documentation refer specifically to the MicroK8s implementation included with Resonate. Your license to use Resonate does not, at this time, include support for running Resonate on other implementations of Kubernetes.

For information on MicroK8s, refer to the MicroK8s documentation at https://microk8s.io/docs.

[1] Currently, your application is responsible for alert monitoring.

[2] This mode is referred to as file-based mode and will be supported in an upcoming release.

Preparing for Installation

This section details the hardware and software requirements and tasks to perform before installing Resonate RFID Reader Management.

Server Preparation Overview

Steps to prepare the machine(s) for Resonate RFID Reader Management installation are outlined below.

The Resonate RFID Reader Management installer includes a prerequisites checker and setup tool, `setup.sh`, which ensures the machine(s) meet these prerequisites.

1. Verify that the machine(s) meet minimum requirements.
2. Create a Zebra SSO account to access the Zebra Artifactory repository and a Zebra Artifactory identity token to run the online installer.
3. Install the operating system with the specified capacities/partitions/volumes.
4. Install the prerequisite software.

   The `setup.sh` script fetches and installs most required software from the internet(online installation) or the setup tar file (offline installation). However, you must install some prerequisite software before you run the script.
5. Configure the machine(s).

After preparing the machines, install Resonate RFID Reader Management.

System Requirements

You can install Resonate RFID Reader Management on a physical or virtual machine(s) that meets or exceeds the following minimum requirements. For a multi-node configuration, each machine must meet these requirements.

These machines should be dedicated to running Resonate RFID Reader Management. System capacities are calculated under the assumption that you will not be running other business applications or services on these computers.

Table 1 System Requirements

Component	Minimum Required
Operating system	Ubuntu Enterprise 22.04 or 24.04 LTS with all the latest patches and updates.
vCPUs	4 vCPUs for single node configuration. 12 vCPUs for multi-node configuration.
Memory (RAM)	24 GB in most cases in a production environment with a maximum of 2000 readers. 16 GB in single-node configuration with fewer than 25 RFID readers.
Extra disk for data storage	128 GB. In single-node configuration, it should be mounted at `/data`; otherwise, it should not be mounted and should not contain partitions or a file system.
Extra partition for /var/snap	50 GB.
Operating system disk	100 GB.

A multi-node configuration sets Resonate in high availability mode. High availability mode requires multiple machines and further requires an odd number of machines. This guide describes how to set up a three (3) machine (node) configuration. It is possible to install a configuration that has more nodes (for example, 5 or 7 or more) to get higher reliability and availability (but not higher capacity). If this is required, contact your Zebra salesperson, who can help you work up a special project and statement of work with Resonate product management and Professional Services.

If you require Resonate alerts sent via email, you must also configure an SMTP server; you need the IP address or hostname of the SMTP server and its port. If you require Resonate alerts sent via REST API to an external case management system, you must configure a webhook URL and login.

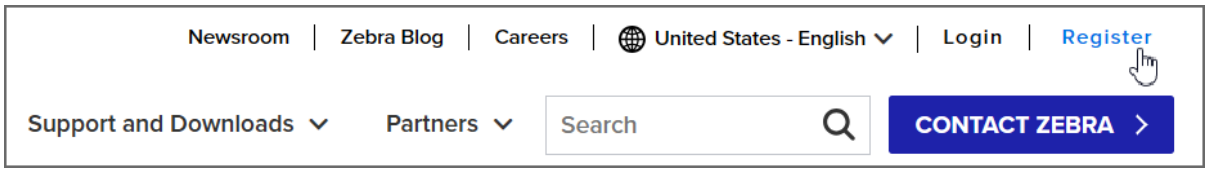
Access to the Zebra Artifactory Repository

To download the Resonate RFID Reader Management software package, create an Zebra SSO account to access the Zebra Artifactory repository. To use the online installer, generate an identity token; use the token as your password when the installer prompts you for your user name and password.

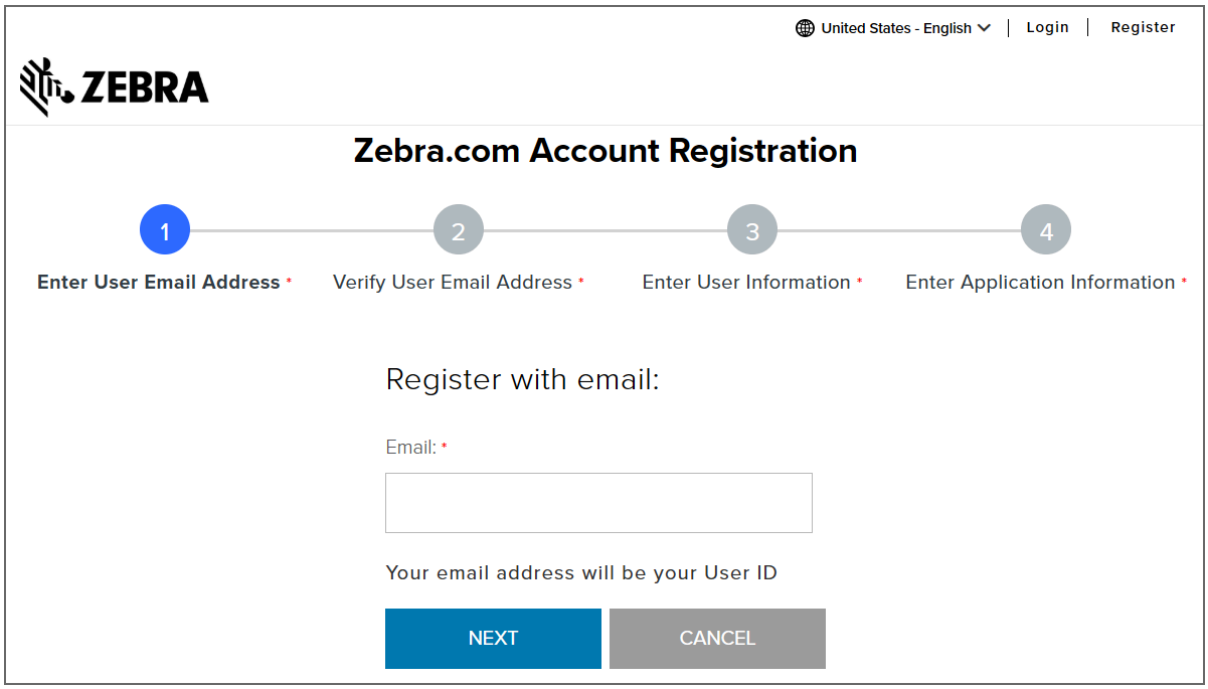
Creating a Zebra SSO Account

The following steps describe how to create a Zebra SSO account so that you can access the Zebra Artifactory repository.

1. Go to www.zebra.com.
2. At the top right of the home page, click **Register**.



3. In the **Email** field, enter a valid email address.
4. Click **NEXT**.



A verification code is sent to the email address.

5. In the **Enter Verification Code** field, enter the code from the email.
6. Click **SUBMIT**.
7. Enter the required information (including password), and accept the terms and conditions.
8. Click **SUBMIT AND CONTINUE**.

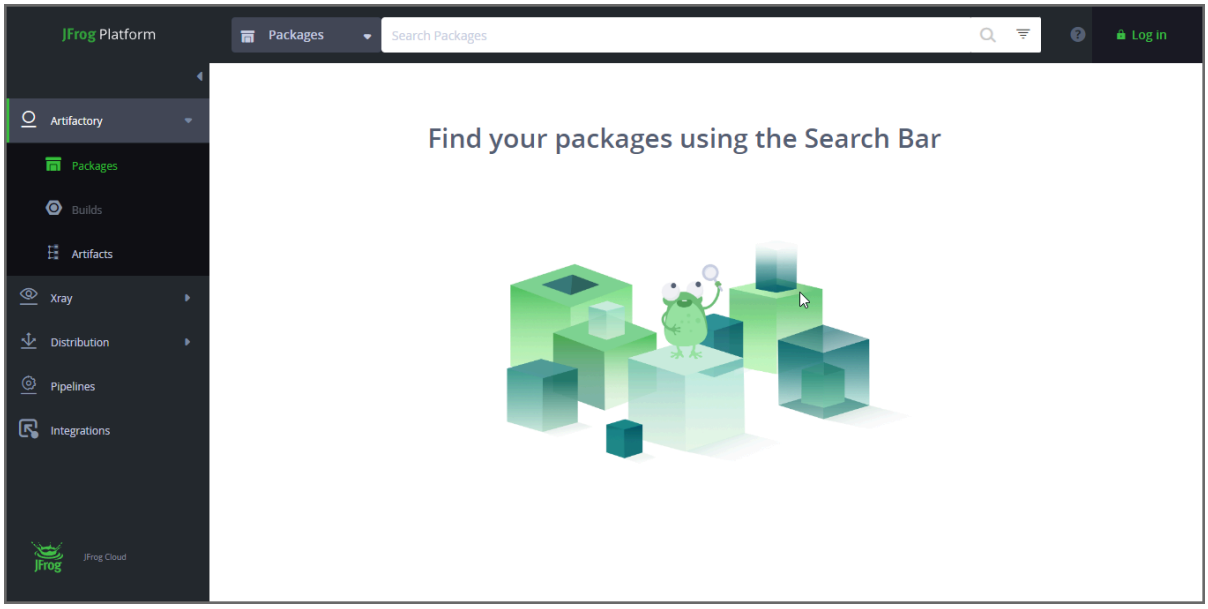
This creates your credentials and returns you to the Zebra home page.

Use these Zebra login credentials to access the Zebra Artifactory repository to download or install Resonate RFID Reader Management software. For online installation, you must also create an identity token.

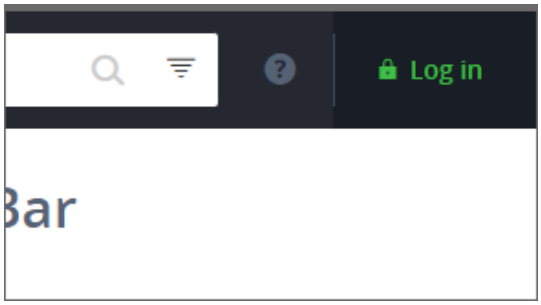
Creating an Identity Token

The following steps describe how to create an identity token required to run the online installer.

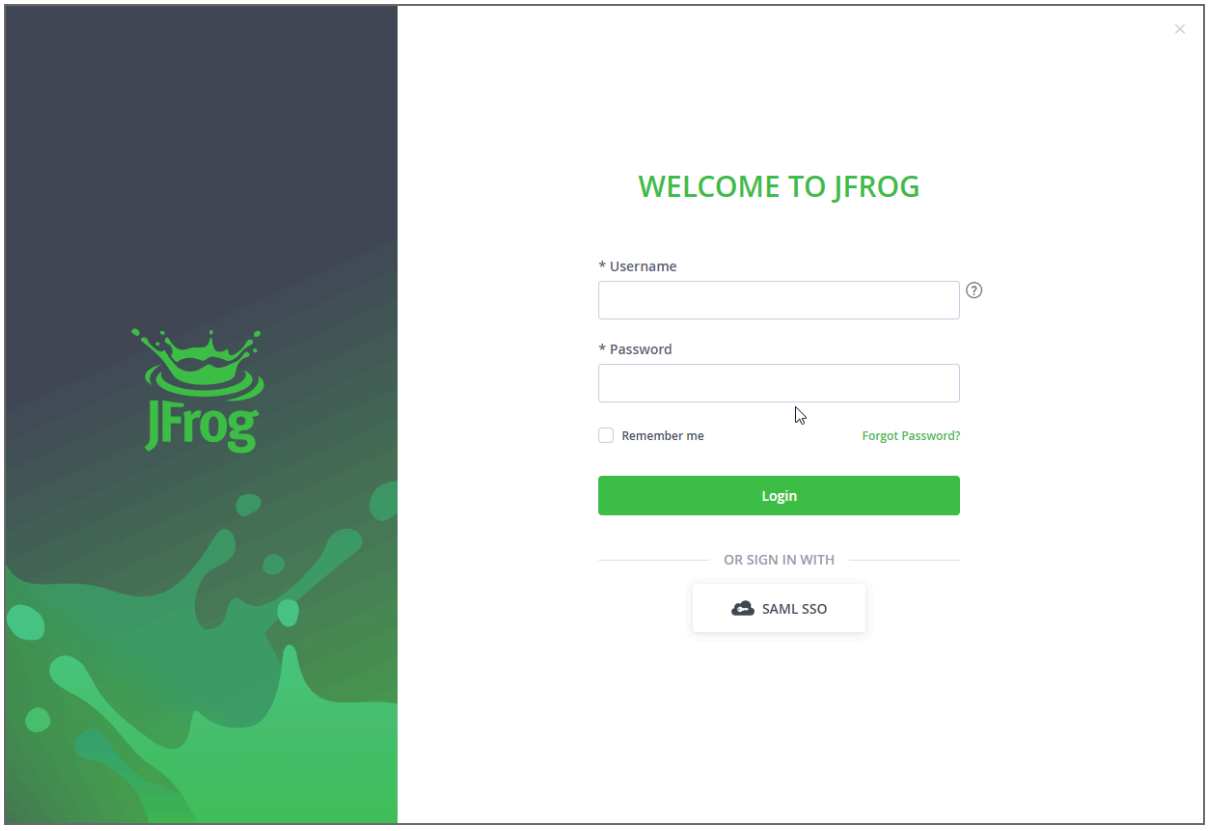
1. Go to zebratech.jfrog.io.



2. Click **Log In**.

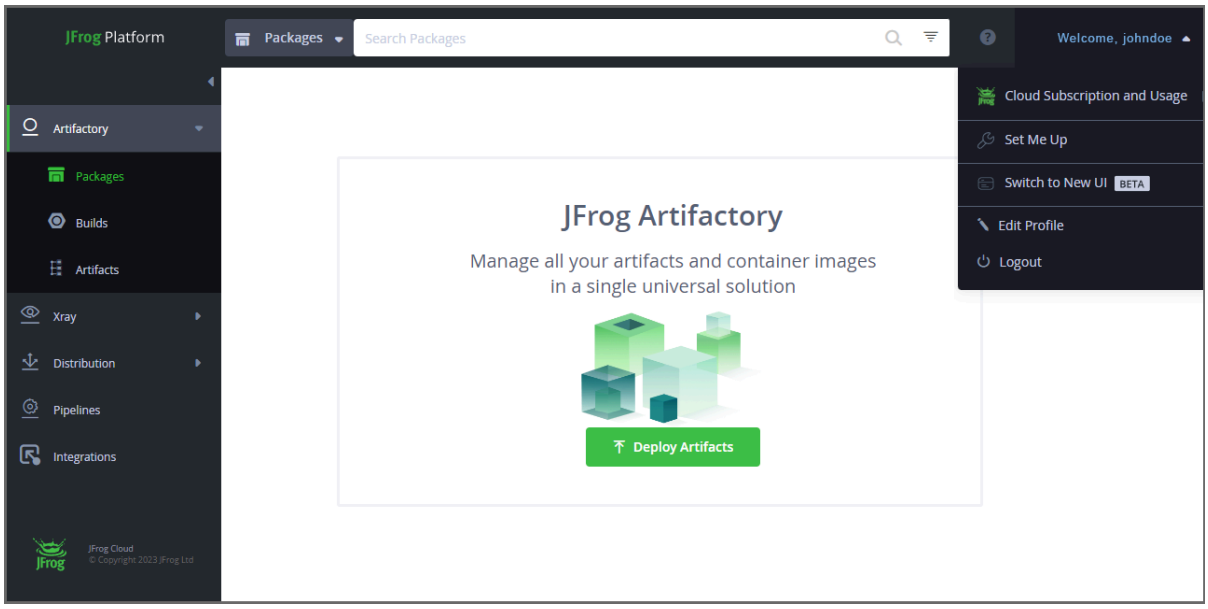


3. Click **SAML SSO**.

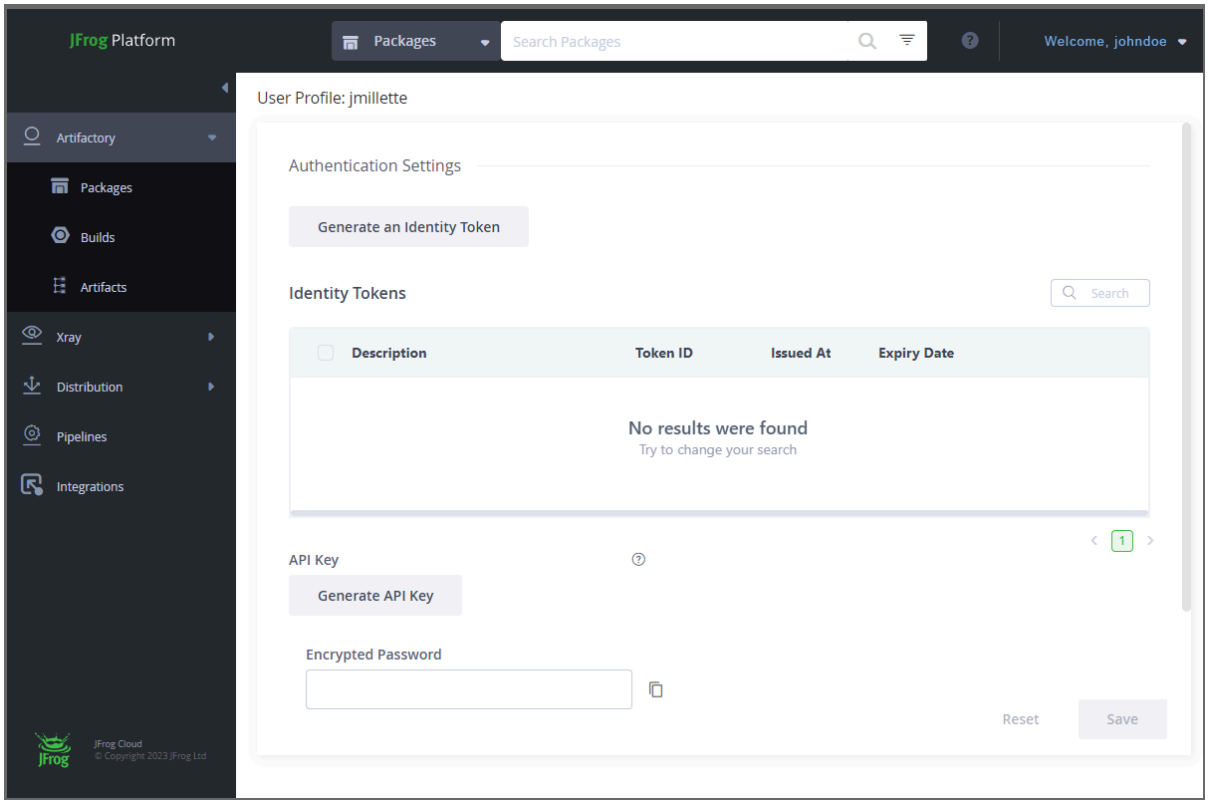


You are logged into the Zebra Artifactory.

4. Click **Welcome, [user name]** > **Edit Profile** at the top right.

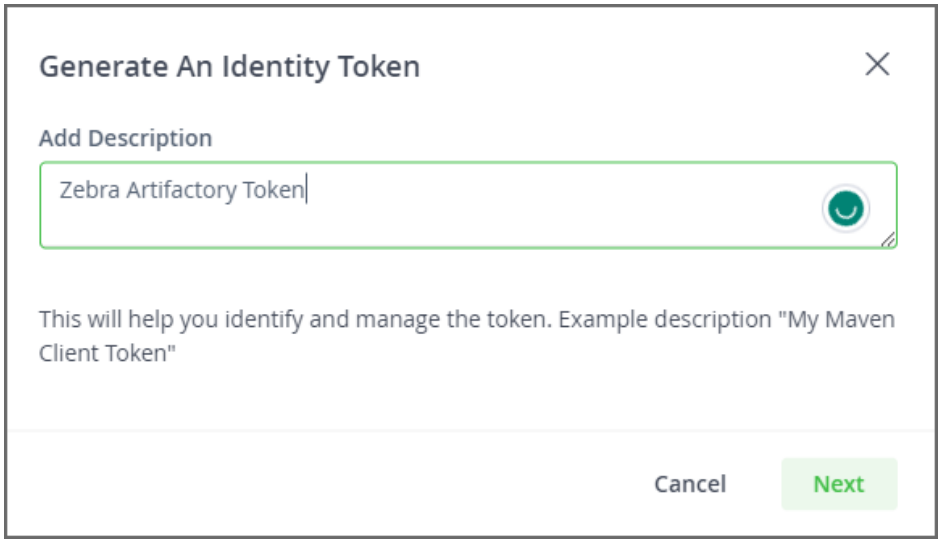


5. Click **Generate an Identity Token**.

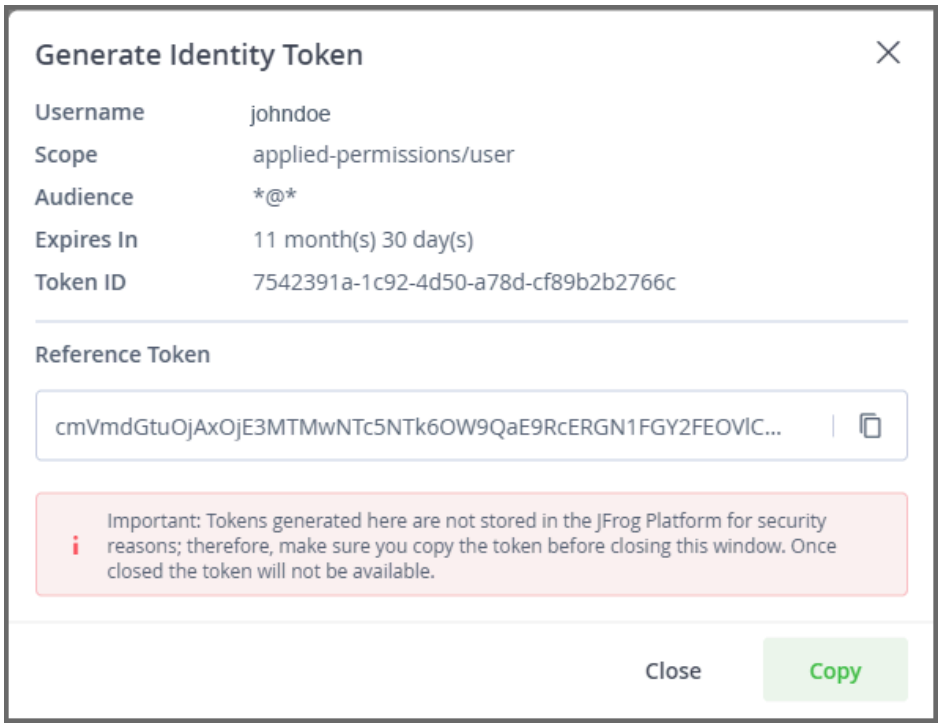


6. In the **Add Description** field, enter a description for the token.

7. Click **Next**.



The information for the identity token is displayed.



8. Click **Copy** at the bottom of the window, or click next to the **Reference Token** to copy the Reference Token text to the clipboard. You can then paste it into a document for reference.

NOTE: It is important to keep a copy of the token. The JFrog Platform does not store the token for security reasons.

When prompted for your user name and password during online installation, use the token as the password.

Prerequisite Software

Resonate RFID Reader Management requires the Snap package manager, several standard Linux utilities for server administration, and the OS Java truststore.

Snap is usually pre-installed on recent versions of Ubuntu. If it is not, install it with the following command:

```
sudo apt update
sudo apt install snapd
```

Ensure the following standard utilities are also installed and updated: `curl`, `findutils`, `coreutils`, and `bash`. To do so, use:

```
sudo apt update
sudo apt install curl findutils coreutils bash
```

Ensure that the Ubuntu OS Java truststore is installed. If it is not, install it with the following command:

```
sudo apt install ca-certificates-java
```

Configuring the Machines

Before installing Resonate RFID Reader Management, you must configure the machines (nodes) that you intend to use. You must set up the Host network settings, time synchronization, DNS, and, if required, SSH. You must also ensure all required network ports are open. The `setup.sh` script of the Resonate installer will create and configure any Resonate-required groups, files/folders, user accounts, and permissions.

Host Setup

Configure the following Host network settings before installing Resonate RFID Reader Management:

- Hostname
- Static IP address (highly recommended), with a valid FQDN
- Subnet mask. Note that for a multi-node deployment, all nodes must use the same subnet mask
- Default gateway

Time Synchronization

The machines require time synchronization with a central Network Time Protocol (NTP) server to maintain consistent system time across the cluster and with the RFID readers. The Resonate server requires an NTP client that synchronizes with the same NTP server as the RFID readers. Install an NTP client if one is not already installed.

DNS

Configure DNS on the machine to enable communication with other Resonate RFID Reader Management components. The DNS server must be able to:

- Resolve the hostname or FQDN of the Resonate RFID Reader Management cluster
- Forward DNS queries to internet DNS servers to resolve public FQDNs (for example, the Zebra Artifactory repository required for an online installation)

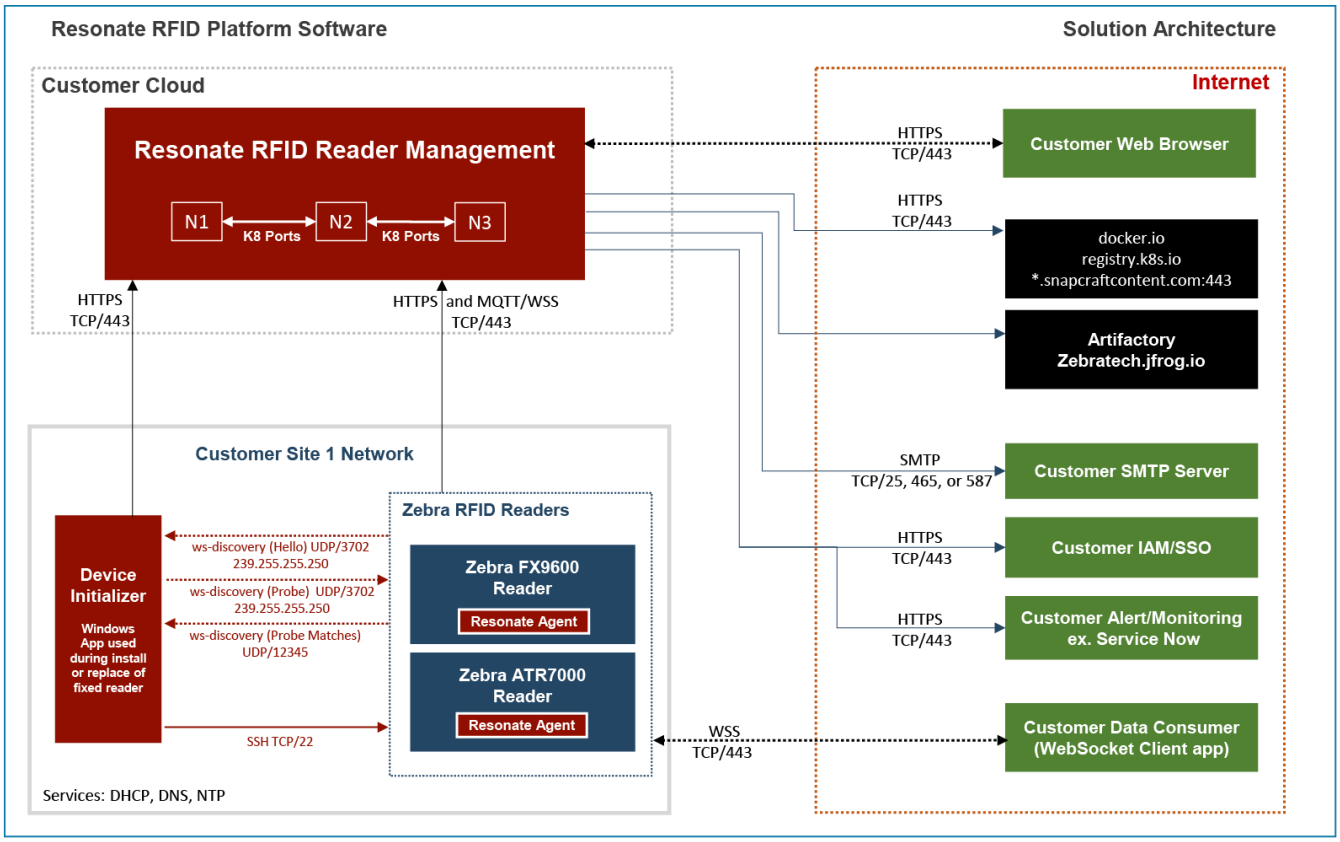
SSH

To run CLI commands securely on the machine, you can set up SSH for remote access, although you can use any other method.

Network Ports

For Resonate software downloads/updates and communication with the RFID readers, the Resonate cluster requires some access permissions and some ports to be open.

Figure 2 Resonate RFID Reader Management Solution Architecture



For an online installation, ensure the server has access to the destination and port shown in the following table. This is the Docker registry that serves all Resonate Service container images.

Table 2 Docker Registry

Destination Host	Destination IP	Destination Port
resonate-doc-rel.artifactory-us.zebra.com	35.201.100.70	443

For Resonate software and RFID reader communication, open the following Resonate server ports.

Table 3 Server Ports

Source	Destination	Protocol and Port number	Description
Passive RFID Reader	Resonate server	HTTPS/TCP port 443	Needed for the Resonate software to deploy, manage, and configure the readers.
SSH client	Resonate server	TCP 22	Needed for an installer or maintainer to get access to the system.
Browser – Resonate web client	Resonate server	TCP 443	Needed for establishing HTTP/HTTPS client connections between the web client and the Resonate server.

For communication between the nodes in a multi-node configuration, open the following ports on each of the machines intended for the cluster; this table is from the MicroK8s documentation at https://microk8s.io/docs. For information on how to open the ports, refer to Microk8s Multi-Node Firewall Documentation; for information about these services, refer to Services and ports in the MicroK8s documentation.

Table 4 Required Ports

Port	Service	Access Restrictions
16443	API server	SSL encrypted. Clients need to present a valid password from a static password file.
10250	kubelet	Anonymous authentication is disabled. X509 client certificate is required.
10255	kubelet	Read only port for the Kubelet.
25000	cluster-agent	Proper token required to authorise actions.
12379	etcd	SSL encrypted. Client certificates required to connect.
10257	kube-controller	Serve HTTPS with authentication and authorization.
10259	kube-scheduler	Serve HTTPS with authentication and authorization.
19001	dqlite	SSL encrypted. Client certificates required to connect.
4789/udp	calico	Calico networking with VXLAN enabled.

Groups

The `setup.sh` script creates the following group: `MicroK8s`. If the group already exists, the script will use it. The script also creates and adds the user `trif-user` to this group.

To give users administrative access to the nodes in the cluster on Linux, add the users to the `MicroK8s` group. This gives the users the same administrative privileges as `trif-user`.

After a multi-node deployment, the group will exist on all the nodes. You only need to add the users to the primary node.

File/Folder Permissions

The `setup.sh` script creates the following files/folders, depending on the deployment, and assigns the required permissions to the user `trif-user`. These files/folders should have the following permissions:

Table 5 Folder and File Permissions

Folder/File	Permission
/opt/zebra/trifecta	drwxr-xr-x. (trif-user:trif-user)
/data/volumes	drwx--x--x. (root:root)

For a multi-node deployment, `/opt/zebra/trifecta` is only needed and created on the primary node; `/data/volumes` is not needed nor created.

User Accounts and Permissions

When creating accounts and adding permissions, your system will automatically have the following two users. Do not add them manually.

Table 6 Account Users

User	Privilege	Group	Description
root	root	`wheel/root`	Service account used to configure the system and install all prerequisites. This user is already present on Linux systems.
trif-user	Resonate service account	`MicroK8s`	Service account used for installation, configuration, and operation of the Resonate software. The `setup.sh` script of the Resonate installer creates and configures the user `trif-user`. In a multi-node deployment, the script only adds the user on the primary node.

MicroK8s Multi-Node Firewall Documentation

This section provides examples showing how to open the required ports (MicroK8s ports) for communication between the nodes in a multi-node configuration. The examples differ in how restrictive and how easy to configure the access is. For more information on the MicroK8s ports, refer to the Services binding to the default Host interface subtopic of the Services and ports topic in the MicroK8s documentation.

Example 1: firewalld - Permitting the Required Ports Between MicroK8s Cluster Nodes

The following shows how to open the required ports (MicroK8s ports) for communication between the nodes in a multi-node configuration. Access to these ports is limited to the nodes in the cluster.

```
node1=10.10.10.10/32
node2=10.10.10.11/32
node3=10.10.10.12/32

sudo firewall-cmd --permanent --new-ipset=MicroK8s-clutser-nodes --
type=hash:ip

sudo firewall-cmd --permanent --ipset=MicroK8s-cluster-nodes --addentry=
$node1
sudo firewall-cmd --permanent --ipset=MicroK8s-cluster-nodes --addentry=
$node2
sudo firewall-cmd --permanent --ipset=MicroK8s-cluster-nodes --addentry=
$node3

sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule source
        ipset=MicroK8s-cluster-nodes port=16443 protocol=tcp accept'
sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule source
        ipset=MicroK8s-cluster-nodes port=10250 protocol=tcp accept'
sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule source
        ipset=MicroK8s-cluster-nodes port=10255 protocol=tcp accept'
sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule source
        ipset=MicroK8s-cluster-nodes port=25000 protocol=tcp accept'
sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule source
        ipset=MicroK8s-cluster-nodes port=12379 protocol=tcp accept'
sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule source
        ipset=MicroK8s-cluster-nodes port=10257 protocol=tcp accept'
sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule source
        ipset=MicroK8s-cluster-nodes port=10259 protocol=tcp accept'
sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule source
        ipset=MicroK8s-cluster-nodes port=19001 protocol=tcp accept'
sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule source
        ipset=MicroK8s-cluster-nodes port=4789 protocol=udp accept'
sudo firewall-cmd --reload
```

To verify that the ports were opened successfully, run the following commands:

```
sudo firewall-cmd --permanent --get-ipsets
sudo firewall-cmd --permanent --ipset=MicroK8s-cluster-nodes --get-entries
```

```
sudo firewall-cmd --permanent --info-ipset=MicroK8s-cluster-nodes
sudo firewall-cmd --zone=public --list-all
sudo firewall-cmd --list-rich-rules
```

Example 2: firewalld - Permitting the Required Ports to All Machines

The following shows how to open the required ports (MicroK8s ports) for communication between the nodes in a multi-node configuration, but in a less restrictive, easier to configure way. This method allows any machine to communicate with these ports rather than limiting access to the nodes in the cluster; typically, other machines should use the external hostname/IP address of the cluster instead of communicating directly with these ports.

sudo firewall-cmd --permanent --zone=public --add-port={ 16443/tcp, 10250/tcp, 10255/tcp, 25000/tcp, 12379/tcp, 10257/tcp, 10259/tcp, 19001/tcp, 4789/udp} sudo firewall-cmd --reload

To verify that the ports were opened successfully, run the following command:

```
firewall-cmd --zone=public --list-all
```

Installing the Software

This section describes how to install the Resonate RFID Reader Management software.

Resonate Software Installation Overview

Resonate RFID Reader Management supports installation in a single-node or multi-node configuration using an online or offline installer. The installer comes with a prerequisites checker and setup tool.

Online Versus Offline Installer

The online installer is lightweight and downloads the Resonate containers from Zebra's Artifactory repository, requiring a direct internet connection during installation; the installer might also download required software from the internet if not already installed.

If you need to install in a protected environment behind a corporate firewall, use the offline installer, which runs without access to the internet or Artifactory; it contains both the Resonate containers and most required software (in case they are missing and require installation).

NOTE: Although the online and offline installers can install most required software, you must install some prerequisite software before you run it. For information, refer to Prerequisite Software.

The installer comes in a setup tar file in the `resonate-<VERSION>.tar.gz` that you receive from Zebra:

- For online installer: `trifecta-installer-k8s-<VERSION>.<BUILD NUMBER>.tar.gz`
- For offline installer: `trifecta-installer-k8s-<VERSION>.<BUILD NUMBER>-offline.tar.gz`

You must extract its files: `setup.sh`, `trifecta.tar`, and `lb.yml`.

Single-Node Versus Multi-Node

You must install Resonate RFID Reader Management according to the required configuration mode: single-node or multi-node.

NOTE: You cannot change between single-node and multi-node after installation.

For a single-node configuration, you run the `setup.sh` script. The setup script creates an installer script, which you then run.

For a multi-node configuration, you run the `setup.sh` script on the primary node, specifying the number of nodes in the cluster and the block device to use for storage. You run the `setup.sh` script on the secondary nodes to prepare them for integration into the cluster, run microk8s on the primary node to get

a token for each secondary node, and then run microk8s on the secondary nodes, passing each a different token. After the nodes have joined the cluster, you run `setup.sh` on each of the secondary nodes to set up their block storage device, and run `setup.sh` on the primary node to finalize the cluster. Finally, you run the installer on the primary node.

Multi-node configuration requires at least 3 nodes initially; you can add more nodes after installation.

Setup

The prerequisites checker and setup tool, `setup.sh`, ensures that the system meets prerequisites, it creates the user `trif-user`, and it extracts the installation files (including `install.sh`) from the `trifecta.tar` file and saves them in the trif-user's home directory (/opt/zebra/trifecta). The `setup.sh` script can fetch and install most required software from the internet (online installation) or the setup tar file (offline installation). However, you must install some prerequisite software before you run the script. For prerequisites, refer to Preparing for Installation on page 10.

Run the `setup.sh` script as root or with sudo. It is typical for `setup.sh` to request that you reboot the system to apply updates. After the reboot, you must run `setup.sh` again, with the exact same options, to continue and complete the setup.

If you are installing a multi-node configuration and you do not have an actual load balancer, run `setup.sh` with the option `--external-ip` (or `-i`) to enable the metalLB load balancer. This allows the nodes to distribute traffic between themselves.

Resonate RFID Reader Management requires a fresh installation and device initialization.

Single-Node Configuration: Setting Up and Installing

The following describes the steps to set up and install Resonate RFID Reader Management in a single-node configuration.

NOTE: When running the online installer, you are prompted for your username and password to access the Zebra Artifactory Repository. As your password, pass the identity token that you generated in Creating an Identity Token on page 13.

1. Ensure that your system meets the prerequisites. Refer to Preparing for Installation on page 10.
2. Extract the files from the setup tar file:
   - For the online installer: `tar trifecta-installer-k8s-<VERSION>.<BUILD NUMBER>.tar.gz`
   - For the offline installer: `tar trifecta-installer-k8s-<VERSION>.<BUILD NUMBER>-offline.tar.gz`

   This extracts `setup.sh`, `trifecta.tar`, and `lb.yml` .
3. Run the following command as `root` or a user with `sudo` privileges:

   ```
   sudo -E ./setup.sh
   ```

   This command ensures that the system meets prerequisites, it creates the user `trif-user`, and it extracts the installation files (including `install.sh`) from `trifecta.tar` into the `trif-user`'s home directory (`/opt/zebra/trifecta`). It displays a success message when it completes successfully.

4. Switch users to the created user trif-user:

```
sudo -E su - trif-user
```

5. Run the following command to begin the installation:

```
./install.sh -h <RESONATE SERVER FQDN> -m prod-single-node --smtp-
server <SMTP SERVER FQDN/IP> --admin-email <EMAIL ADDRESS>
```

   Replace <RESONATE SERVER FQDN> with the FQDN of the server, <SMTP SERVER FQDN/IP> with the FQDN or IP address of the SMTP server, and <EMAIL ADDRESS> with the administrator's email address for receiving notifications. For additional script information and options, refer to install.sh on page 35.

   The installation sets up the necessary Kubernetes resources (such as pods, services, and deployments) required for Resonate RFID Reader Management to operate. It can take anywhere from 10 to 20 minutes. It displays `Installation complete` when it completes successfully and Resonate RFID Reader Management is operational.

6. Verify that Resonate RFID Reader Management is operational. Refer to Validating That Resonate Is Operational on page 40.

When the Resonate software is installed and operational, its services are running; you can now access and use the software through the Resonate web interface. You should receive an email at the specified administrator address with a web link to the newly installed Resonate RFID Reader Management platform (at the FQDN that you provided during installation), requesting you to set the administrator's password.

If your SMTP server uses a custom Certificate Authority (CA), you must first configure Keycloak to trust the certificate on your SMTP server before you receive the email; for information, refer to Keycloak and Using a SMTP Server with Custom CA on page 29.

Multi-Node Configuration: Setting up and Installing

The following describes the steps to set up and install Resonate RFID Reader Management in a multi-node configuration. You should perform them as `root` or a user with `sudo` privileges.

NOTE: When running the online installer, you are prompted for your username and password to access the Zebra Artifactory Repository. As your password, pass the identity token that you generated in Creating an Identity Token on page 13.

1. Ensure that all systems intended for the cluster meet the prerequisites for a multi-node configuration, especially ensuring that their K8s ports permit access to and from the other nodes in the cluster. Refer to Preparing for Installation on page 10.
2. On all the systems, extract the files from the setup tar file:
   - For the online installer: `tar trifecta-installer-k8s-<VERSION>.<BUILD NUMBER>.tar.gz`
   - For the offline installer: `tar trifecta-installer-k8s-<VERSION>.<BUILD NUMBER>-offline.tar.gz`

   This extracts `setup.sh`, `trifecta.tar`, and `lb.yml` .

Primary Node - Initial Setup

3. On the primary node, run the following command:

```
sudo -E ./setup.sh -m <N> -b <DEVICE> [-i <IP>]
```

Replace <N> with the number of nodes intended for the cluster and <DEVICE> with the path of the block device for storage. If you do not have a load balancer (bare metal), use the -i option and replace <IP> with a unique external IP address to use for the cluster, on the same subnet as the nodes, and with a DNS record; this enables the `metalLB` load balancer. For example, to set up a 3-node cluster with free storage on /dv/nvme1n1 and without a preexisting load balancer, using 10.10.10.13 as the IP of the `metalLB` load balancer, the command would be:

```
sudo -E ./setup.sh -m 3 -b /dev/nvme1n1 -i 10.10.10.13
```

For additional information and options, refer to setup.sh on page 34.

This command ensures that the system meets prerequisites, it creates the user `trif-user`, and it extracts the installation files (including `install.sh`) from `trifecta.tar` into the `trif-user`'s home directory (/opt/zebra/trifecta). It displays a success message when it completes successfully.

Secondary Nodes - Setup

4. On each secondary node, run the following command:

```
sudo -E ./setup.sh --microk8s-only
```

This installs only microk8s on the system.

5. On the primary node, run the following command as many times as there are secondary nodes:

```
sudo -E /snap/bin/microk8s add-node
```

This command returns a token command to run on one of the secondary nodes. You cannot reuse this token command on all secondary nodes; each node requires a unique one. Take note of the token command that each call returns.

6. On each of the secondary nodes, run a different token command from the previous step (obtained from the primary node). It should look similar to the following:

```
sudo -E /snap/bin/microk8s join <IP OR HOSTNAME>:25000/<TOKEN>
```

There is no need to wait for a node to join the cluster before adding the next node. You can add them in parallel.

7. On each secondary node, after it joins the cluster, run the following command to set up its storage:

```
sudo -E ./setup.sh --storage-only -b <DEVICE>
```

Replace <DEVICE> with the path of the block device of the secondary node. This generally resembles `/dev/` which might be `/dev/sdb`, `/dev/nvme1n1`, or something similar.

8. On the primary node, run the following command to finalize the setup of the cluster:

```
sudo -E ./setup.sh --finalize
```

Installation

9. On the primary node, switch users to the created user `trif-user`:

```
sudo -E su - trif-user
```

10. Run the following command to begin the installation:

```
./install.sh -h <RESONATE SERVER FQDN> -m prod-multi-node --smtp-
server <SMTP SERVER FQDN/IP> --admin-email <EMAIL ADDRESS>
```

    Replace <RESONATE SERVER FQDN> with the external hostname (FQDN) of the server (cluster), <SMTP SERVER FQDN/IP> with the FQDN or IP address of the SMTP server, and <EMAIL ADDRESS> with the administrator's email address for receiving notifications. If using your own load balancer, set the FQDN of the Resonate server to the FQDN of the load balancer. For additional script information and options, refer to install.sh on page 35.

    The installation sets up the necessary Kubernetes resources (such as pods, services, and deployments) required for Resonate RFID Reader Management to operate. It can take anywhere from 10 to 20 minutes. It displays `Installation complete` when it completes successfully and Resonate RFID Reader Management is operational.

11. Verify that Resonate RFID Reader Management is operational. Refer to Validating That Resonate Is Operational on page 40.

When Resonate is installed and operational, its services are running; you can now access and use the software through the Resonate web interface. You should receive an email at the specified administrator address with a web link to the newly installed Resonate RFID Reader Management platform (at the FQDN that you provided during installation), requesting you to set the administrator's password.

If your SMTP server uses a custom Certificate Authority (CA), you must first configure Keycloak to trust the certificate on your SMTP server before you receive the email; for information, refer to Keycloak and Using a SMTP Server with Custom CA on page 29.

Starting Resonate RFID Reader Management

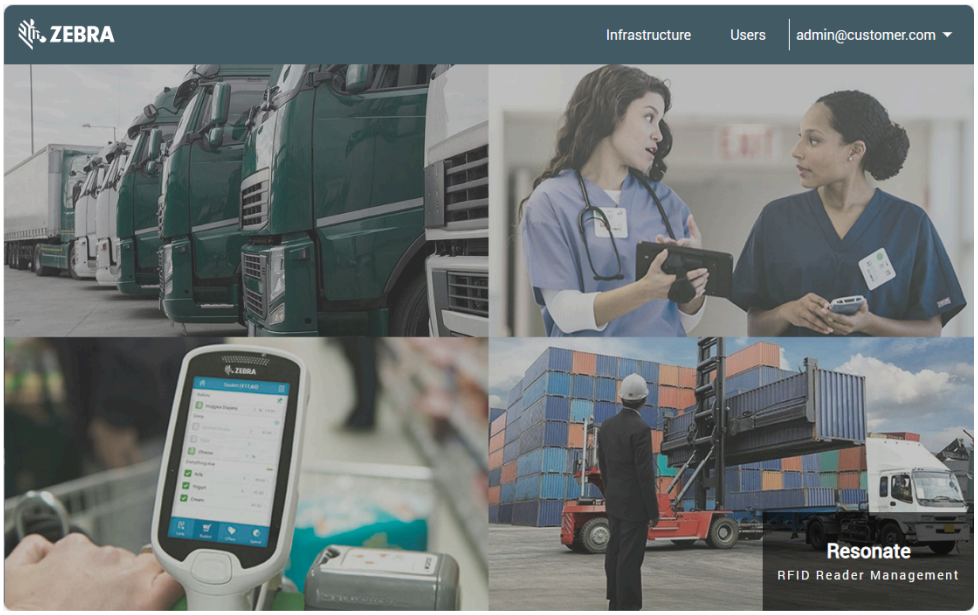
When Resonate RFID Reader Management is installed successfully, the system automatically starts all the services.

Resonate RFID Reader Management User Interface

To interact with Resonate RFID Reader Management, use its web user interface (UI).

It is accessible from a browser at:

```
https://<Fully Qualified Domain Name of Resonate Server>/
```



Login using the administrator credentials specified during installation.

Resonate Device Initializer and Initializing the RFID Readers

After installing Resonate RFID Reader Management, you must run the Resonate Device Initializer utility on a Windows machine to initialize the readers; this also installs the Resonate Agent on the readers. The utility is separate from Resonate so that it can run locally on the same network segment as the readers for discovery purposes.

Resonate Device Initializer supports two modes:

- Discovery-based device onboarding: Resonate Device Initializer automatically discovers all supported RFID readers on the same subnet. In this case, you must run the utility onsite and on the same subnet as your RFID readers.
- File-based device onboarding[3]: Resonate Device Initializer requires that you provide a file with an explicit list of supported RFID readers that you need Resonate RFID Reader Management to manage. You can run the utility from any Windows machine provided there is network access to the RFID readers.

The Resonate Device Initializer utility is a self-contained executable (`rm-device-initializer-<VERSION>.exe`) without an installer. It is distributed in the `resonate-<VERSION>.tar.gz` file that you received from Zebra.

Resonate requires a fresh installation and device initialization.

For information on how to use the utility, refer to the Resonate RFID Reader Management User Guide.

[3] File-based mode will be supported in an upcoming release.

Adding a Node

Without reinstalling Resonate RFID Reader Management, you can add a machine (node) to a multi-node configuration (cluster).

NOTE: You cannot add a node to a single-node cluster.

Follow the instructions in the Secondary Nodes - Setup section of Multi-Node Configuration: Setting up and Installing on page 25.

Only perform the steps for the node that you are adding. Ensure to run the `--finalize` step from the primary node at the end.

Removing a Node

Without reinstalling Resonate RFID Reader Management, you can remove a machine (node) from a multi-node configuration (cluster) as outlined below. If the node is inaccessible, refer to Removing an Inaccessible Node From the Cluster instead.

NOTE: You cannot remove the primary node. In addition, a multi-node cluster cannot have fewer than three nodes.

1. On the node to remove, run the following command:

```
microk8s leave
```

2. On the primary node, run the following command and establish the name of node to remove:

```
kubectl get nodes
```

   You can perform this step before or after performing the previous step.

3. On the primary node, run the following command:

```
microk8s remove-node <NODENAME>
```

   Replace <NODENAME> with the name of the node to remove.

The node is no longer part of the cluster.

Keycloak and Using a SMTP Server with Custom CA

In an environment where your Simple Mail Transfer Protocol (SMTP) server uses a custom Certificate Authority (CA), configure Keycloak to trust the certificate on the SMTP server.

You must do this after installing Resonate RFID Reader Management and before you can access it. If Keycloak is not configured to trust the certificate, you will not receive the email at the specified administrator address with the link to the newly installed Resonate RFID Reader Management platform, requesting you to set the administrator's password.

To trust the certificate, add it to the Keycloak truststore, by default, located at `/etc/ssl/certs/java/cacerts`. This is the standard location for the OS Java truststore on Ubuntu. If this truststore already contains your custom CA's certificate, Keycloak already trusts your certificate. Otherwise, add the

certificate to this truststore using a tool like Keytool. Keytool requires that you set a password. Remember this password for later; you need to specify it when you add the certificate to the Keycloak truststore.

If you change the location of the Keycloak truststore, make sure it is accessible to `trif-user` on the cluster. `/tmp` will most likely work.

Adding the Certificate to Keycloak

For Resonate to trust the certificate on your SMTP server, you must add the certificate to Keycloak.

1. Add the certificate to the Keycloak truststore, by default, located at `/etc/ssl/certs/java/cacerts`.

   Use a tool like Keytool to add the certificate to the truststore. Keytool requires that you set a password. Remember this password for later; you need to specify it when you add the certificate to the Keycloak truststore.

2. Create a ConfigMap using the certificate files:

```
kubectl create configmap certificates --from-file=cacerts=path/to/your/
certs
```

3. Mount this ConfigMap into the Keycloak pod. To do so, edit the Keycloak deployment:

```
kubectl patch deployment keycloak -p '{"spec": {"template": {"spec":
{"volumes": [{"name": "certificates", "configMap": {"name":
"certificates"}}], "containers": [{"name": "keycloak", "volumeMounts":
[{"name": "certificates", "mountPath": "/etc/ssl/certs/java/
cacerts"}]}]}}}'
```

4. If using a custom truststore, set the password for the truststore in the Keycloak deployment:

```
 kubectl patch deployment keycloak -p '{"spec": {"template": {"spec":
{"containers": [{"name": "keycloak", "env": [{"name": "JAVA_OPTS",
 "value":
"-Xms64m -Xmx512m -XX:MetaspaceSize=96M -XX:MaxMetaspaceSize=256m -
Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.err.encoding=UTF-8
 -
Dstdout.encoding=UTF-8 -Dstderr.encoding=UTF-8 -XX:+ExitOnOutOfMemoryError
 -
Djava.security.egd=file:/dev/urandom -XX:+UseParallelGC -
XX:MinHeapFreeRatio=10 -XX:MaxHeapFreeRatio=20 -XX:GCTimeRatio=4 -
XX:AdaptiveSizePolicyWeight=90 -
Djavax.net.ssl.trustStorePassword=password"}]}]}}}'
```

   The password is at the end of the environment variable. Replace password with the password that you set when creating the truststore.

After you have added the certificate to your trust store, configure the SMTP settings in Keycloak. Refer to Configuring the SMTP Settings in Keycloak on page 31.

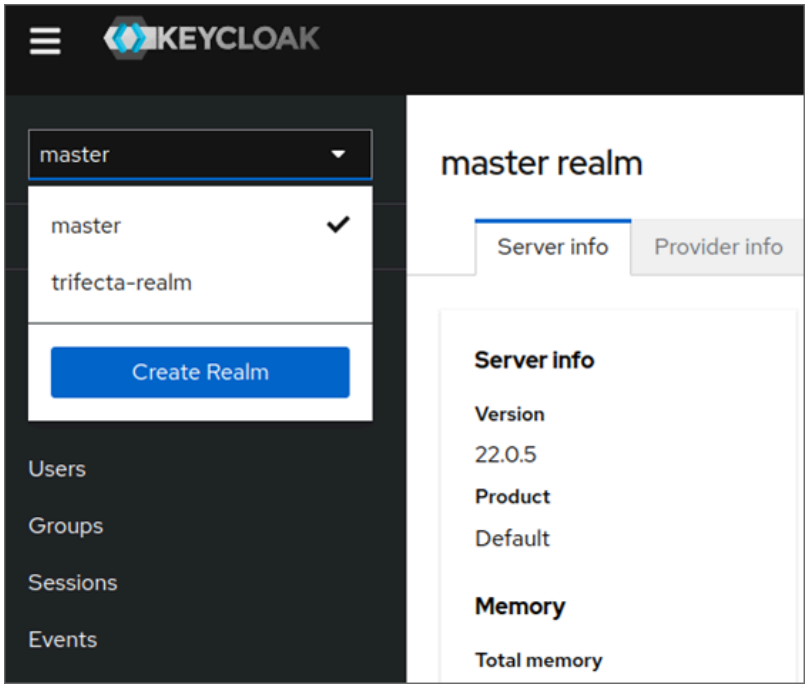
Configuring the SMTP Settings in Keycloak

After you have added the certificate to your trust store, configure the SMTP settings in Keycloak.

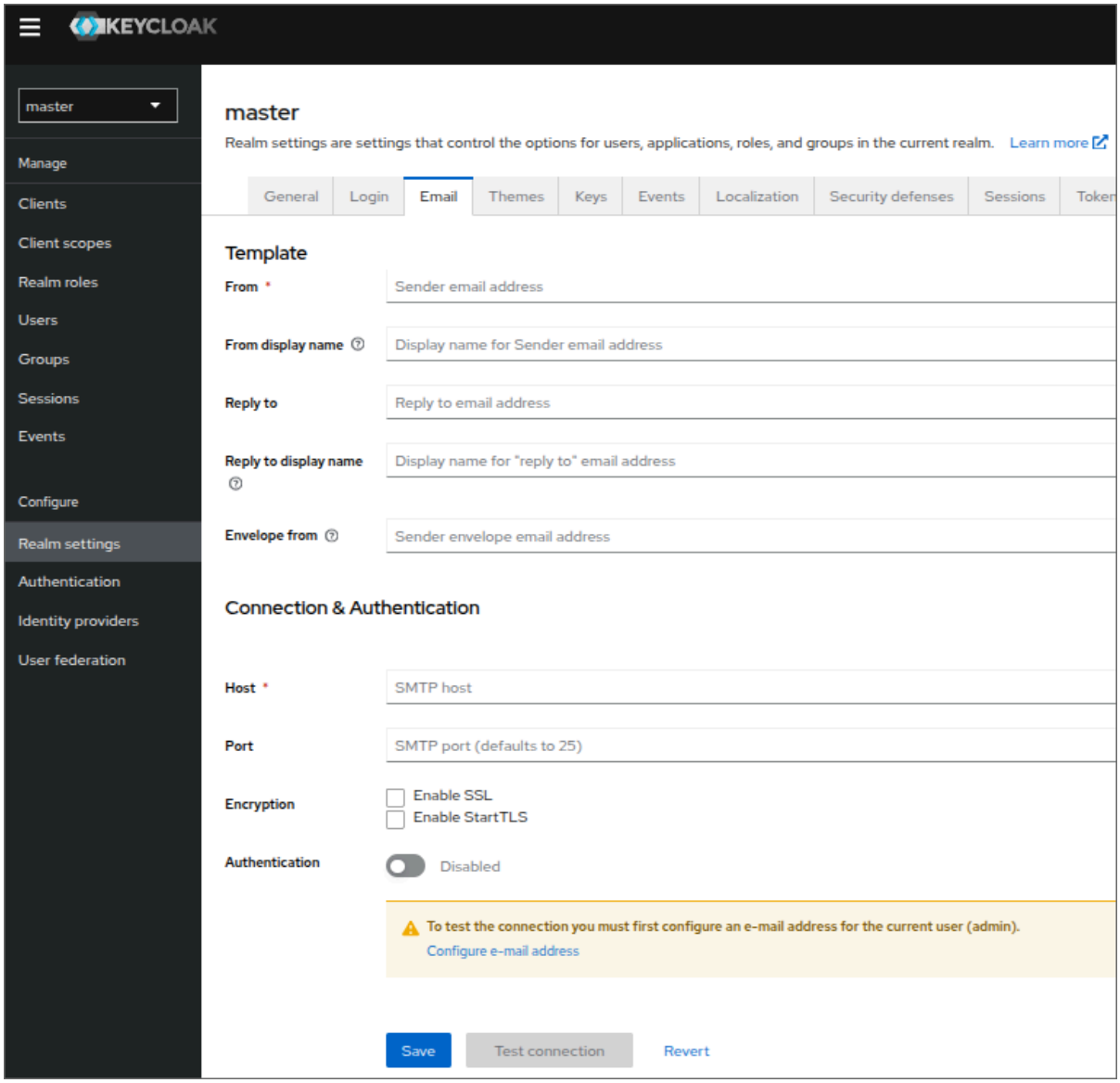
1. Retrieve the Keycloak admin password. Extract this from the cluster using the following command:

```
kubectl get secrets keycloak-initial-admin -o jsonpath='{.data.password}'
 | base64 -d && echo
```

2. Log in to the Keycloak administration console at `https://<RESONATE FQDN SERVER>/trifecta/v1/keycloak/admin/` with the username `admin` and the password retrieved in the previous step.
3. At the top left of the landing page, set the realm to `trifecta-realm`.



4. Click the **Realm Settings** tab on the left, and then select the **Email** tab in the page presented.

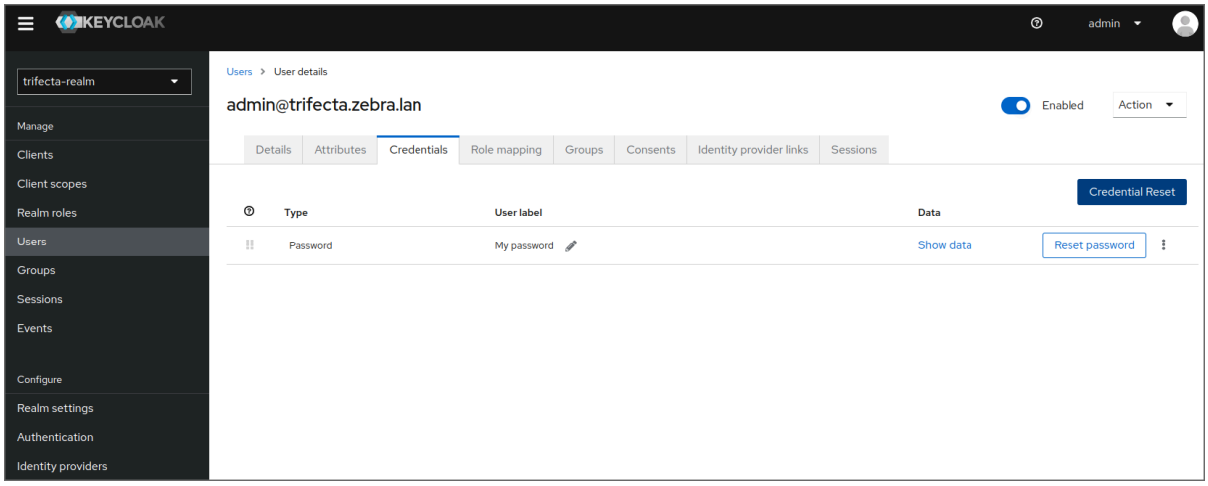


5. Set the following SMTP settings:

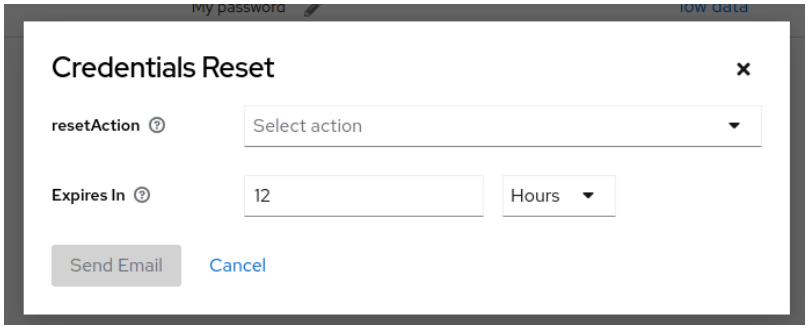
   - **From**: Specify the same SMTP account as the one you specified during installation.
   - **Host**: Specify the SMTP hostname. The installation might have already set this.
   - **Port**: Specify the appropriate port. For the default SMTP server, specify 25. If you are using SSL or STARTTLS, specify 465 or 587, respectively.

   If you require authentication, you should probably use SSL or STARTTLS. You must set the **Username** field to `admin` and **Password** field to the password retrieved using keycloak-initial-admin.

6. Click **Test Connection** to verify connectivity. You might need to set the email for the admin account in the master realm. Alternatively, you can navigate to the **Users** tab and select your user.



7. Click **Credentials** and then **Credential Reset**.



This sends an email to the specified user to reset their password. You receive an error message if SMTP fails to send.

8. Check the Keycloak logs for additional details:

```
kubectl logs -l app=keycloak
```

Installation Scripts

This section serves as a reference, with a detailed description of the Resonate RFID Reader Management scripts utilized during installation.

setup.sh

The `setup.sh` script is a prerequisite checker and sets up your system(s) for Resonate RFID Reader Management installation. It has several command-line options to customize the installation process, particularly for use for a multi-node cluster.

Command Syntax

```
setup.sh [options]
```

Options

Option	Description
`--external-ip=<IP>` `-i <IP>`	Specifies the external IP address of the cluster. The address must be unique, use the same subnet as the nodes, and have a DNS record. The readers and the Resonate web interface use this IP address to access the cluster. When you specify an external IP address, it activates the `metallb` load balancer in an environment where a load balancer is not provided, allowing the nodes to distribute traffic between themselves. Default: N/A Required: For a multi-node setup.
`--multi-node=<NODES>` `-m <NODES>`	Specifies the number of nodes in the cluster. This determines the number of replicas for storage. Default: 1 Required: For a multi-node setup.
`--block-device=<DEVICE>` `-b <DEVICE>`	Specifies the path to the block device to use for storage. Default: N/A Required: For a multi-node setup.

Option	Description
`--microk8s-only` `-k`	Specifies to install only microk8s on the node. This prepares the node for integration into the cluster, allowing it to be added (bootstrapped) as a secondary node using the `microk8s add-node` command from the primary node. Specify this option alone. Default: N/A Required: No
`--storage-only` `-o`	Specifies to only install storage. Use this option on secondary nodes after they join the cluster to specify their storage device. Specify this option alone. Default: N/A
`--finalize`	Specifies to finalize the joining of the nodes to the cluster. Use this option on the primary node after joining all nodes to the cluster and setting up their storage device. Specify this option alone.
`--help` `-h`	Displays the setup's possible options and their descriptions (this table).

install.sh

The `install.sh` script installs Resonate RFID Reader Management. You run the script after setting up the system(s) with the `setup.sh` script. In a multi-node configuration, run the script from the primary node. `install.sh` has several command-line options to customize the installation process.

Command Syntax

```
install.sh [options]
```

Options

Option	Description
`--external-hostname=<RESONATE SERVER FQDN>` `-h <RESONATE SERVER FQDN>`	Specifies the external hostname of the server (cluster). This is the FQDN to use to access the server (cluster) from outside the network. Default: N/A Required: Yes.
`--mode=<MODE>` `-m <MODE>`	Specifies the deployment mode for installing the server. Valid `<MODE>` values are: • `prod-single-node` • `prod-multi-node` Default: prod-single-node

Option	Description
	Required: For a multi-node setup.
`--smtp-server <SMTP SERVER FQDN/IP>`	Specifies the FQDN or IP address of the SMTP server to use. Default: N/A Required: Yes.
`--smtp-port <PORT>`	Specifies the port of the SMTP server to use. Default: 25 Required: No
`--admin-email <EMAIL>`	Specifies the email address to use for the system administrator (first user with administrator privileges). When the system is up, the installer sends an email to this address with a link to the newly installed platform, requesting that the user set their password. Default: N/A Required: Yes.
`--tls-certificate <TLSCERTIFICATE>` `-c <TLSCERTIFICATE>`	Specifies the path on your Resonate server to your trusted X.509 digital enterprise certificate. Provide a file that includes the full certificate chain; the `install.sh` script assigns the certificate to the server. Use this option to avoid the `self-signed certificate` error when browsing to the Resonate web interface. Before using this option with the installer, add the certificate to the new Resonate server's file system; the certificate should be PEM encoded. If required, you should typically run `install.sh` with this option at Resonate installation time or before adding readers; otherwise, you will have to re-initialize the readers. When passing this option, also use the `--tls-key <TLSKEY>` option to pass the private key. Default: N/A Required: No
`--tls-key <TLSKEY>` `-k <TLSKEY>`	Specifies the TLS private key associated with the X.509 digital enterprise certificate. Default: N/A Required: No
`--tls-override`	Overrides the existing TLS certificate and private key in the cluster. Default: N/A

Option	Description
	Required: No
`--help`	Displays the installer's possible options and their descriptions (this table). Default: N/A

Validation

This section provides information to validate your system requirements and that your Resonate software is operational.

System Validation

Resonate RFID Reader Management requires certain system specifications to function properly and automatically creates the `trif-user` user and the `microk8s` group.

Table 7 Server Configuration

Resource	Minimum Requirement	Validation Command
OS	Ubuntu Enterprise 22.04 or 24.04 LTS.	`more /etc/os-release`
Memory	24 GB in most cases. 16 GB in single-node configuration without many RFID readers.	`lsmem | grep "Total online memory"`
vCPU	4 vCPUs for single node configuration. 12 vCPUs for multi-node configuration.	`lscpu | grep "CPU(s)"`
Disk layout and sizing	Extra disk for data storage: 128 GB. In single-node configuration, it should be mounted at `/data`; otherwise, it should not be mounted and should not contain partitions or a file system. Extra partition for `/var/snap`: 50 GB Operating system disk: 100 GB.	If mounted, use: `df -h -x overlay` To see all available disks and partitions, whether mounted or not, use either: • `lsblk` • `sudo fdisk -l`
Hostname	Verify the domain is appended.	`hostname -f`

Table 7 Server Configuration (Continued)

Resource	Minimum Requirement	Validation Command
Resonate user	The `trif-user` user is in the `microk8s` group. The setup.sh script of the Resonate installer automatically creates the user and group.	`groups <trif-user administrative user>`

Table 8 Network Configuration

Resource	Validation Command
DNS name resolution for all systems	Run the nslookup command for each hostname and ensure that each resolves:
	`nslookup <hostname>`
	`<hostname> = Docker host`
	`<hostname> = NTP server`
Time is in sync with the NTP server	Verify time synchronization using:
	`date`
	or
	`chronyc sources`

NOTE: Commands and output can vary. If Chrony is not used as the NTP service, instead of using `chronyc sources`, use the command associated with your NTP service to verify that time is in sync with the NTP server.

Table 9 MicroK8s Configuration

Resource	Validation Command	Validation Result
MicroK8s	`microk8s status`	Reports information such as the status of core MicroK8s services and components and the enabled/disabled add-ons.
	`kubectl get nodes`	In a multi-node configuration, it reports information such as: • Node status (`Ready` or encountering issues) • Version of Kubernetes currently running • Current node's role (for example, master) • Any issues or warnings

Validating That Resonate Is Operational

The following steps describe how to ensure Resonate RFID Reader Management is operational.

1. Run the following command to switch to the user `trif-user` account:

```
sudo su – trif-user
```

2. Run the following command to list the status of all Resonate RFID Reader Management pods and ensure that they are running:

```
kubectl get pods -n zebra-reader-management
```

If Resonate RFID Reader Management is operational, its pods should display a `Running` or `Completed` status.

```
trif-user@resonate-rm:~$ kubectl get pods -n zebra-reader-management
NAME                                    READY   STATUS      RESTARTS      AGE
alert-enricher-69ff66c49d-nvq5j         1/1     Running     2 (26d ago)   26d
alerting-sidecar-575f8cdf74-5qbq5       1/1     Running     3 (26d ago)   26d
broker-dual-role-0                      1/1     Running     0             26d
broker-entity-operator-7d549df94c-q89b2 2/2     Running     0             26d
device-manager-7475fb8b45-7g7kz         1/1     Running     0             26d
file-manager-cb7fd6fb-vrhrz             1/1     Running     0             26d
iotc-adapter-fixreader-98f7f7977-m69vs  1/1     Running     5 (26d ago)   26d
keycloak-0                              1/1     Running     0             26d
keycloak-db-1                           1/1     Running     0             26d
keycloak-kube-proxy-75c95bbdff-2hdfj    1/1     Running     0             26d
kube-node-cleanup-gk2hr                 1/1     Running     0             26d
kube-secret-cleanup-29250360-9tv4w      0/1     Completed   0             66m
kube-secret-cleanup-29250390-kt52r      0/1     Completed   0             36m
kube-secret-cleanup-29250420-lrgm5      0/1     Completed   0             6m42s
map-service-6f6bb4dbc-6gttp             1/1     Running     0             26d
minio-pool-0-0                          2/2     Running     0             26d
mqtt-core-854b85b85d-0                  1/1     Running     0             26d
prometheus-kafka-bridge-b569d8474-vsjc6 1/1     Running     0             26d
tileserver-777587db76-fq4ps             1/1     Running     0             26d
trifecta-pooler-678ffb5695-nxktc        1/1     Running     0             26d
trifecta-postgres-1                     1/1     Running     0             26d
ui-devices-66fd7cf9f7-r47p2             2/2     Running     0             26d
ui-host-7769c567fc-dg4xn                2/2     Running     0             26d
ui-sites-759dd49d5b-6qr55               2/2     Running     0             26d
version-service-6dc89866c-zk5jj         1/1     Running     0             26d
trif-user@resonate-rm:~$
```

If there is a `grafana` pod and it reports `ContainerCreating`, ignore it.

Troubleshooting

This section describes solutions to possible issues during installation. It also lists various troubleshooting commands.

Kubernetes

The MicroK8s troubleshooting documentation provides a comprehensive guide to troubleshooting common issues with MicroK8s. To access the documentation, go to https://microk8s.io/docs/troubleshooting. This section covers general troubleshooting when using Kubernetes. This is intended as a guide to help determine the state of the system and resolve common issues that might occur.

IMPORTANT: Only make changes if you are confident that it will not impact the overall operations of the system. If you need assistance, contact Zebra support.

Generic Troubleshooting Commands

These are typical commands you will use when viewing the basic status of Kubernetes. Each command has numerous options. This document goes into further details, with more specific commands as needed.

Command	Description
`kubectl -h`	Prints the help for kubectl.
`kubectl get pods`	Prints a table of pod information in the default namespace.
`kubectl get pods -o wide`	Prints additional information showing the node on which the pods are running.
`kubectl get pods -n`	Prints a table of pod information in a specific namespace.
`kubectl describe pods <pod name>`	Outputs the pod configuration and events.
`kubectl logs <pod name>`	Outputs the logs for the container within the pod. If multiple containers are within a pod, it will show the logs for the default container.
`kubectl logs <pod name> -c <container name>`	Outputs the logs for a specific container within the pod. Use this when there are multiple containers within a pod and logs are needed for a specific container.

Command	Description
`kubectl get nodes`	Prints a table of node information.
`kubectl get namespaces`	Prints a table of all namespaces.

Kubernetes Status

This section describes how to get the pod status for the default namespace, the pod status for storage, and the node status.

Pod Status For Default Namespace

Check the status of application pods within the default namespace. All pods should show as ready and `Running`.

```
kubectl get pods -n default
```

The output displays a table of all pods.

Table 10 Pod Status

NAME	READY	STATUS	RESTARTS	AGE
<pod name>	*<n>/<n>*	`Running`	*<number of restarts>*	*<age>*

The following explains each of the columns in the output.

Table 11 Pod Status Columns

Column	Description
NAME	The unique name of the pod.
READY	The number of containers in the pod that are ready. For example, 1/1 indicates that all containers in the pod are ready and running, whereas 1/2 indicates that only 1 is running.
STATUS	The current execution of the status (such as Pending, Running, Succeeded, Failed, Completed or Unknown).
RESTARTS	The number of times the Pod has restarted. It will indicate the pod's stability and the occurrence of any issues that might have initiated the restart.
AGE	The time when pod was created.

The following explains common pod STATUS states.

Table 12 Pod Status States

State	Description
`Pending`	The pod has been accepted by the Kubernetes cluster, but one or more of the containers has not been set up and made ready to run. This includes time a pod spends waiting to be scheduled as well as the time spent downloading container images over the network. If the node is stuck in this state, generally this is because there are insufficient resources (CPU/Mem), or because the images are being downloaded. The output of `kubectl describe <pod name>` should provide messages from the scheduler about why it can not schedule the pod.
`Running`	Pod is actively running on a node, with at least one container running.
`Succeeded`	All containers in the pod have terminated successfully and will not be restarted.
`Failed`	One or more containers in the pod have terminated with a non-zero exit status, or a container failed to restart.
`Unknown`	Pod state is not known to the controller, usually because of a communication issue between the node and the Kubernetes API.
`ContainerCreating`	If a pod gets stuck with this message, a common cause is that an image did not fully download. To get more information about this state, run: `kubectl describe pod <pod name> -n <namespace>` If it is not fully downloaded, wait. If the problem does not resolve itself, contact Zebra support. If there is grafana pod and it reports ContainerCreating, ignore it.
`Init:N/M`	Pod status `Init:N/M` means one init container is not finalized; `init:N/M` means the pod has `M` Init containers, and `N` have completed so far. To get more information, use the following commands: • `kubectl describe pods` • `kubectl logs` • `kubectl logs -c init-container-xxx`
`CreateContainerConfigError`	When the configuration specified for a container in a pod is not correct or is missing a vital part. For example, there is a missing persistent volume, ConfigMaps, or secret. You can attempt to determine the cause by describing the pod, with the command: `kubectl describe pod -n <namespace> <podname>`
`CreateContainerError`	This is a problem happening at a later stage in the container creation flow. Kubernetes displays this error when it attempts to create the container in the pod.

Pod Status for Storage

If you are using multi-node deployments, check the status of storage pods in the `rook-ceph` namespace. All pods should show as ready and `Running`, with the exception of `rook-ceph-osd-prepare`, which will be `Completed`.

```
kubectl get pods -n rook-ceph
```

Add the `-o wide` option in the `kubectl get pods` command above to show the pod status and which pods are running on individual nodes.

Node Status

Check the status of nodes. All nodes should show `Ready`.

```
kubectl get nodes
```

The output displays a table of all nodes.

Table 13 Node Status

NAME	STATUS	ROLES	AGE	VERSION
<node name>	`Ready`	none	*<age>*	*<version>*

The following table explains each of the columns in the output.

Table 14 Node Status Columns

Column	Description
NAME	The name of the node within the cluster.
STATUS	The current status of the node, indicating whether it is ready, not ready, or unreachable.
ROLES	Any roles assigned to the node, such as master, worker, etc.
AGE	The duration since the node was created or added to the cluster.
VERSION	The Kubernetes version is running on the node.

The following table explains the node STATUS states.

Table 15 Node STATUS States

State	Description
`Ready`	Healthy node.
`NotReady`	Node down or unreachable. This can be caused by a communication issue or node failure.
`Ready, SchedulingDisabled`	Node cordoned. This is the outcome of running `kubectl cordon` or `kubectl drain`, which prevents a node from scheduling new pods. Outside of maintenance, this is typically not desired. To uncordon (tell the node to resume scheduling), run `kubectl uncordon`.

Troubleshooting Kubernetes

This section covers troubleshooting when using Kubernetes itself.

Besides using the MicroK8s troubleshooting documentation, use the following information to troubleshoot kernel issues.

Table 16 Troubleshooting the Kernel

Problem	Cause	Solution
Error displayed in logs of the Kubelet service (by default, snap.microk8s.daemonkubelite. service): `open /proc/sys/ net/netfilter/ nf_conntrack_max: no such file or directory`	On occasion, the conntrack kernel module might not load. You can then verify if the module is not loaded by running: `lsmod | grep conntrack` If the module is not loaded, you will not see any output.	Load the conntrack module manually: `sudo modprobe nf_conntrack`

Use the following information to troubleshoot general Kubernetes issues.

Table 17 Troubleshooting Kubernetes

Problem	Cause	Solution
When the Kubernetes cluster is under disk pressure, you might see large quantities of pods stuck in the Error, Evicted, or ContainerStatusUnknown states.	This is because the kubelet is unable to create new pods due to the lack of disk space.	Free up disk space on the node (for example, delete unused images), or increase the disk space available to the node.
		Run the following command to clean up all pods: `kubectl delete pod --all-namespaces --field-selector=status.phase=Failed`
		Verify the error is caused by disk pressure by checking the output of the node description: `kubectl get node -o 'jsonpath={.status.conditions[?(@.type=="DiskPressure")].status}'` This returns True if the node is under disk pressure, and False otherwise.

Table 17 Troubleshooting Kubernetes (Continued)

Problem	Cause	Solution
A node has joined the cluster but has communication issues. When a node attempts to rejoin, the following error occurs: `Connection failed. The joining node (IP address of failed remote node) is already known to dqlite (504).`	The node might be in `NotReady` state.	Remove the failed node by running the command: `sudo -E /snap/bin/microk8s remove-node --force <NODE NAME>` Re-add the node, follow the instructions in the Adding a Node on page 29 . Only perform the steps for the node that you are re-adding. Run the `--finalize` command from the primary node at the end.
You have an incorrect FQDN, or you are unable to log in to the application.	If you install with an incorrect FQDN, you won't be able to log in to the application.	Uninstall Resonate RFID Reader Management with the command: `sudo -E su - trif-user ./uninstall.sh` The script is in the same folder as install.sh.
		Install Resonate RFID Reader Management with the command: `./install.sh -h <RESONATE SERVER FQDN> -m <MODE> --smtp-server <SMTP SERVER FQDN/IP> --admin-email <EMAIL ADDRESS>` Refer to Installing the Software on page 23 and install.sh on page 35
Containers fail to download or start: Pod STATUS state is: `ErrImagePull`	A container running in a pod fails to pull the required image from a container registry. This error can occur for a variety of reasons, including network connectivity issues, incorrect image name or tag, missing credentials, or insufficient permissions.	Check network connectivity, verify the image name and tag, ensure that credentials are correct and up to date, and ensure that the appropriate permissions are in place. Once the issue is resolved, the pod can be recreated, and the container image should be pulled successfully.

Table 17 Troubleshooting Kubernetes (Continued)

Problem	Cause	Solution
Containers fail to download or start: Pod STATUS is: `ImagePullBackOff`	The result of repeated ErrImagePull errors.	
Containers fail to download or start: Pod STATUS is: `CreateContainerConfigError`	Kubernetes can not create the container based on the configuration.	If the pod is stuck in this state, contact Zebra for additional support.
Containers fail to download or start: Pod STATUS is: `CreateContainerError`	Kubernetes can not create the container, but not due to the configuration.	Review container logs. Fix the issue being reported.
Pods are stuck in `Pending` state.	This is often due to a storage issue.	Check the state of your PersistentVolumeClaims (PVCs). Refer to Resolving Storage Failures on page 47

Resolving Storage Failures

While installing the software, you might find that some of your pods are stuck in the `Pending` state. This is often due to storage issues.

Check the state of your PersistentVolumeClaims (PVCs):

```
kubectl get pvc
```

If the PVCs show a status of `Pending`, describe the PVC to see the reason:

```
kubectl describe pvc <pvc name>
```

Under the `Events` section, you will see the reason for the PVC being stuck in the `Pending` state.

You might see that the error is due to the storage controller not being able to create the volume. This could be due to an issue with the storage controller.

If you are using a multi-node deployment with `Rook`, you can check the status of the storage pods in the `rook-ceph` namespace:

```
kubectl get pods -n rook-ceph
```

All pods should show as either `Completed`, or `Running and ready`.

If the storage pods are not running, you can check the logs for the pods to see if there are any errors:

```
kubectl logs <pod name> -n rook-ceph
```

The operator `pod rook-ceph-operator-*` frequently logs errors that can help you identify the issue.

Troubleshooting Network Flow

If you are experiencing network flow issues, first verify that there is successful communication from the client to the cluster before troubleshooting the cluster and applications within the cluster. Network flow issues include connectivity issues, latency and slow performance, packet loss, unresponsive network devices, or unexpected drops or resets in the connection.

NOTE: Client-to-cluster network communication is critical to the operation of the overall product. Typical network configurations, such as IP addressing, routing, DNS, and firewalls, are some of the basic components expected to be operational and are outside of the scope of this document. Before troubleshooting the cluster and applications within the cluster, first verify that there is successful communication from the client to the cluster.

After ensuring there is successful communication from the client to the cluster, troubleshoot the cluster and applications. Traffic in the cluster involves multiple layers, generally resembling the following:

1. Load balancer.
2. Ingress controller.
3. Kubernetes service.
4. Pod and containers.

You can follow traffic through these components to identify connecting issues. You can also use the returned HTTP status code to assist in troubleshooting.

Table 18 Troubleshooting Network Flow Issues

Problem	Cause	Solution
HTTP status codes: 502, 503, or 504	It is likely that the issue lies with the Kubernetes service or pod responsible for processing requests.	Depending on the route, you should be able to identify if the pod that is trying to be reached is not running or is not responding. `kubectl get pods -n zebra-reader-management` All application pods should show a status of `Running` and a ready count of N/N.
Connectivity issues, latency, slow performance, packet loss, unresponsive network devices, or unexpected drops or resets in the connection.	Traffic may not be reaching the cluster.	Verify that traffic is reaching the cluster. Refer to Following Request Logs

Following Request Logs

To verify that traffic is reaching the cluster, look at the logs for the ingress controller. Depending on the number of nodes, you might need to check multiple logs. The logs show the incoming request and the response.

```
kubectl logs -n ingress nginx-ingress-<extra>
```

You should see messages similar to the following:

```
<Node IP> - - [<Timestamp>] "<HTTP Method> <Path> HTTP/2.0" <Response Status
 Code> <Content Length> "-" "User Agent" 68 0.010 [Backend] [] <Backend IP/
Port> 81947 0.010 200
```

Use this to determine if the ingress controller was able to connect to the service. If it was, you can then check the logs for the service to see if it received the request.

```
kubectl logs -n zebra-reader-management <service pod name>
```

If the service logs show that the request was received, you can also check for errors or exceptions, which might have caused the issues that you are seeing.