Swimlane (10x) Platform Installer Guide

1. Swimlane Platform Installer Guide

When you need to access the SPI UI after the initial install you can access it by access port 8800 over HTTPS on any node IP (e.g. https://:8800).

> All configuration of the Swimlane Platform Installer and Swimlane platform must be done through the SPI admin console config page. Editing or manipulating the underlying Kubernetes resources is not supported and will not be permanent since they are managed and controlled by the Swimlane Platform Installer.

Configure the Swimlane Platform for an Embedded Cluster Install

To configure the Swimlane platform for an embedded cluster install:

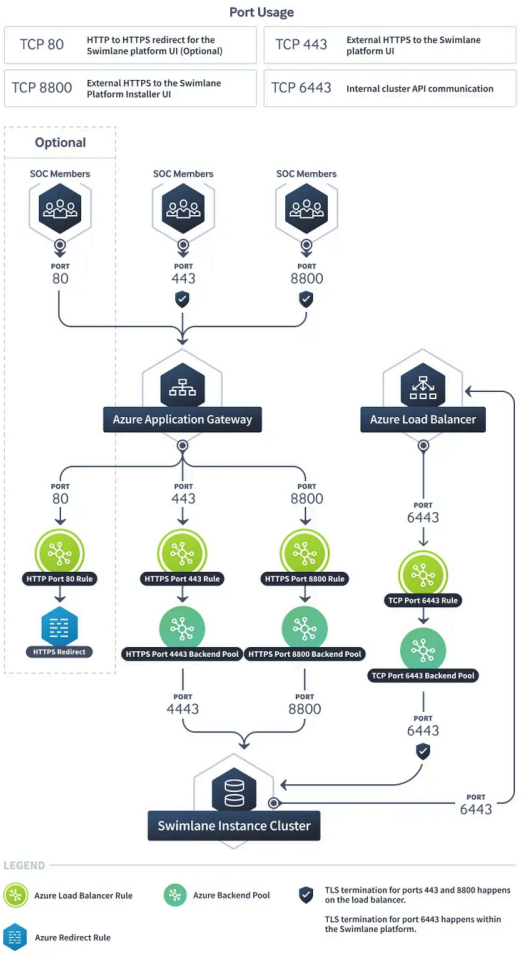
Swimlane Settings (Ingress & Web)

- On Swimlane Settings, you'll begin setting up your configuration for Swimlane. Review and set the following fields as necessary:
    - **Expose the Swimlane Web Service Externally?** - Enable this option when using a Layer 7 load balancer. The Swimlane web service will be directly exposed from each node in the cluster on TCP port 4443.
    - **Enable the Ingress Controller** - Enable this option when using a Layer 4 load balancer or for single node lab/test environments. The included ingress controller will be used for routing web requests to Swimlane on TCP port 443.
        - **Swimlane Hostname** - The DNS record pointing to the Swimlane Platform Installer and Swimlane platform load balancer.
        - **Upload a certificate for Swimlane Web?** - Enable this option to upload a certificate and key to be used for the included ingress controller. If no certificate is uploaded a self-signed one will be generated and used. The certificate must be ASCII encoded X.509 format. The private key cannot be password protected.

1.1.2.2.5. Azure Standard_v2 Application Gateway

This topic explains how to use an Azure Standard_v2 Application Gateway (Layer 7) for your Swimlane deployment. Two load balancers are required for this deployment. The Azure Standard_v2 Application Gateway (Layer 7) is used for external access to the Swimlane platform and the Swimlane Platform Installer. An additional Azure Load Balancer (Layer 4) is still required for the internal cluster communication.

Architecture Diagram



Standard_v2 Application Gateway for the Swimlane Platform and the Swimlane Platform Installer

- Create an **Azure Standard_v2 Application Gateway**
  - `Resource Group` should be set according to your organization's standards
  - `Region` should match that of the Virtual Machines that Swimlane will be installed

1.1.4. Add Additional Nodes to Create an HA Cluster

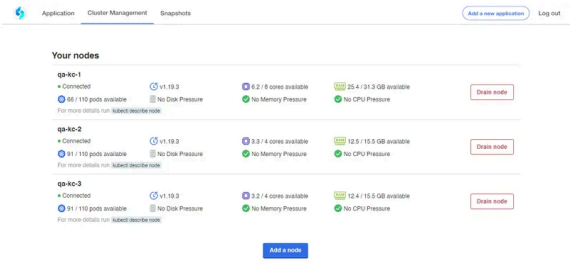
This topic provides instructions on how to create a High Availability (HA) cluster and assumes you have already run the Swimlane Platform Installer on the first node. Please see **Install Into an Embedded Kubernetes Cluster** if you have not yet installed on the first node.

HA considerations

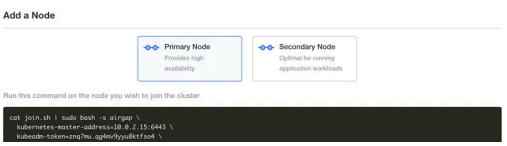
- HA clusters require at least 3 nodes and the total number of nodes must be an odd number.
- Only one MongoDB pod can run on a node. For example, you need 3 nodes in a cluster if you want to go to 3 MongoDB pods to enable a replica set.
- The first node that the initial install is running on should be in the target group to start. The additional nodes should only be added to the target groups after they've been joined to the cluster.

To add additional nodes to a cluster:

1. From the Swimlane Platform Installer UI, click the Cluster Management tab.



2. The Cluster Management page displays your installed nodes.
3. Click **Add a node**.
4. Under Add a Node, ensure that you have selected **Primary node** and then click **Copy command**.



1.1.12.2. Automate an Offline Embedded Installation

This topic provides automation steps for setting up an offline, single-node Swimlane installation on embedded clusters.

Installation Preparation and Customization

Before you begin, review the **System Requirements for an Embedded Cluster Install** to confirm your system's compliance. The Swimlane Platform Installer (SPI) performs several checks in the initial phases of the installation to ensure the underlying host is compatible with the application. To bypass any applicable checks, an installer patch YAML file can be applied at installation time to account for several of these settings. Refer to **Overriding Installer Settings** for more options.

Installer Patch File Example

Here is an example installer patch file:

```
apiVersion: "cluster.kurl.sh/v1beta1"
kind: "Installer"
metadata:
  name: "patch"
spec:
  kubernetes:
    HACluster: true
    loadBalancerAddress: "<Load Balancer FQDN or IP>:6443"
  firewalldConfig:
    disableFirewalld: true
  selinuxConfig:
    disableSelinux: true
```

You also need the following files:

- Swimlane Platform Installer license file (ends with `.yaml`)
- Swimlane license file (ends with `.lic`)
- Swimlane tar archive (ends with `.tar.gz`)
- Swimlane airgap bundle (ends with `.airgap`)

1.2.7. Backup and Restore on an Existing Cluster with Snapshots

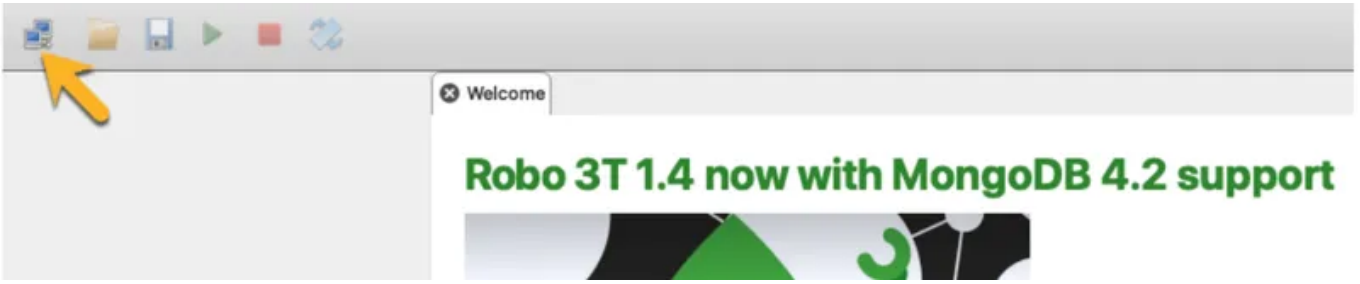
1.3.2. Installing Robo 3T and running a query for MongoDB

Step 1: Installing Robo 3T

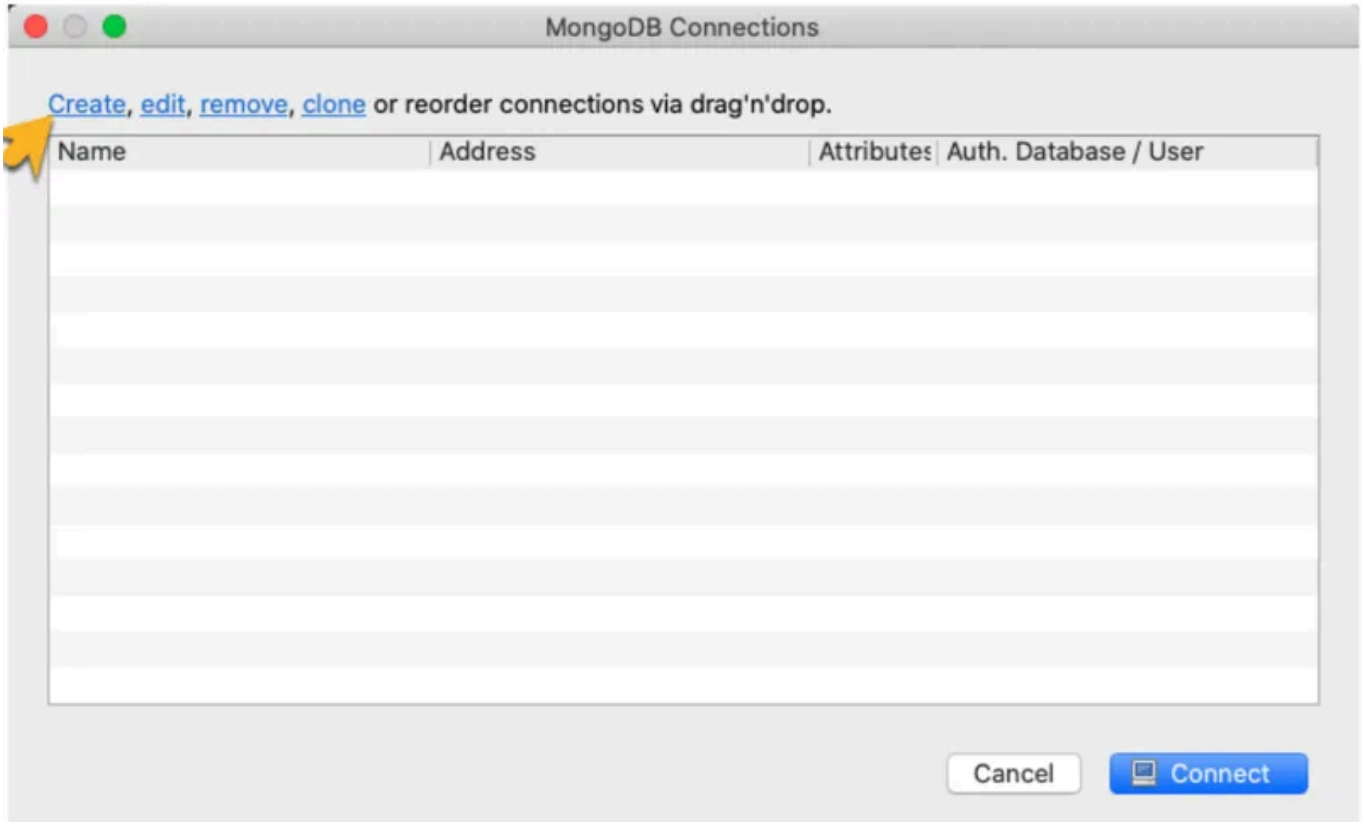
1. Download Robo 3T version on your respective OS (we are using Mac OS for this article) from https://robomongo.org/download.

Step 2: Setting up and Connecting to a single host Linux CentOS system

1. Click on the two computers at the upper left-hand corner as shown in the screenshot below:



2. This will open the connection window then click on create as shown below:



3. Click the connection tab:

1.3.12. Using Velero commands for SPI Backup and Restore

Velero is a tool for managing disaster recovery, specifically for Kubernetes (K8s) cluster resources. It provides a simple, configurable, and operationally robust way to back up your application state and associated data. If you're familiar with kubectl, Velero supports a similar model, allowing you to execute commands such as 'velero get backup' and 'velero create schedule'. The same operations can also be performed as 'velero backup get' and 'velero schedule create'. Instead of using the SPI admin UI to do backup and restore, one can also use velero CLI to achieve the same.

Create SPI backup

```
velero create backup <BackupName> OR <BackupID>
```

Schedule a backup:

```
velero create schedule <name> --schedule [flags]
Examples:
# Create a backup every 6 hours
velero create schedule NAME --schedule="0 */6 * * *"
# Create a backup every 6 hours with the @every notation
velero create schedule NAME --schedule="@every 6h"
# Create a daily backup of the web namespace
velero create schedule NAME --schedule="@every 24h" --include-namespaces web

# Create a weekly backup, each living for 90 days (2160 hours)
velero create schedule NAME --schedule="@every 168h" --ttl 2160h0m0s
```

Troubleshooting a failed backup:

Please use velero debug --backup <backupname> to generate the support bundle, and attach to the ticket, more options please refer to velero debug --help

Sample:

```
velero debug  --backup instance-2hg5z
2023/06/28 05:00:19 Collecting velero resources in namespace: velero
2023/06/28 05:00:23 Collecting velero deployment logs in namespace: velero
2023/06/28 05:00:31 Collecting log and information for backup: instance-2hg5z
```

1.3.22. How to Reduce the Restic Prune Interval to Retrieve Deleted Velero Backup Disc Space

When deleting velero backups, the 'delete backup' commands are asynchronous: they are queued and executed later by restic like garbage collection. The default restic prune interval is seven days, so it can take up to seven days before a velero backup is actually deleted from disc.

To retrieve disc space from deleted backups more quickly, you can force restic to prune 'delete backup' commands within five to ten minutes by executing:

```
kubectl -n velero get resticrepositories -oname \
  | xargs -I{} kubectl -n velero patch {} \
  --type='json' \
  -p '[{"op":"replace", "path":"/status/lastMaintenanceTime", "value":"2020-
01-01T00:00:00Z"}]'
```

1.3.23. Resetting Swimlane Platform Installer (SPI) Console Admin Password

Once the admin password is stored it cannot be read back, the only way to recover is to reset it.

```
kubectl kots reset-password <namespace>
```

This is also valid syntax to change the password:

```
kubectl kots reset-password -n <namespace>
```

NOTE: for embedded clusters the namespace is default

1.3.24. How to Work Around SPI Installer Incompatible Version Problem

1.3.32. kubectl commands are failing due to proxy

Kubectl commands may fail with the following error:

```
E0403 14:04:05.944359 3116197 memcache.go:265] couldn't get current server
API group list: Get "https://swimlane.dev.swimlane.example.com:6443/api?
timeout=32s": Service Unavailable
```

1. Check if there is a proxy enabled in the environment

   ```
   env | grep -i proxy
   ```

   The resulting output should be as below if there is a proxy enabled in place:

   ```
   https_proxy=http://192.168.x.x:8080
   http_proxy=http://192.168.x.x:8080
   ```

2. Disable proxy from the node

   ```
   export
   no_proxy=127.0.0.1,192.168.x.x,.default,kubernetes,.local,localhost,.svc,
   load balancer fqdn
   ```

   The above should also be added to **$HOME/.bashrc** otherwise a node reboot will wipe it.

3. Check to make sure the proxy is disabled.

   ```
   env | grep -i proxy
   ```

   The output from above should now be as below.

   ```
   https_proxy=http://192.168.x.x:8080
   http_proxy=http://192.168.x.x:8080
   no_proxy=127.0.0.1,192.168.x.x,.default,kubernetes,.local,localhost,.svc,
   load balancer fqdn
   ```

4. kubectl commands should work now.

1.3.33. Upload a New TLS Certificate