SevOne Data Insight 6.6 Administration Guide

24 August 2023
IBM SevOne NPM Version 6.6.0
Document Version 6.6.1.0

Table of Contents

SevOne Documentation

All documentation is available from the IBM SevOne Support customer portal.

1 About

SevOne Data Insight, SevOne's new reporting framework, provides you with real-time insight and analytics across diverse and high-scale environments. SevOne Data Insight enables you to leverage rich, interactive, and responsive dashboards with reports and analytics for high-impact insight into your network and service delivery infrastructure.

This document contains instructions on how to perform administrative tasks such as, setup, configuration, and deployment for SevOne Data Insight.

IMPORTANT: PLEASE READ BEFORE CONTINUING...

As of SevOne Data Insight 3.5 release, the operating system has been changed from **Red Hat Enterprise Linux (RHEL)** to **CentOS 7**. You will not be able to rollback the operating system once you install SevOne Data Insight 3.5 or higher version.

Warning# 1

SevOne Data Insight depends on SevOne NMS to generate the reports. It is very important that resource names in SevOne NMS that are being used by SevOne Data Insight are NOT changed. Changing the resource names will break the reports that are dependent on them.

Warning# 2

New installation of SevOne Data Insight can be performed by only using **.iso** / **.ova** that is the matching version of SevOne Data Insight. Using an older version of the **.iso** / **.ova** file to provision a newer version of SevOne Data Insight is not supported due to minimum platform dependencies requirements.

Please do not run **sevone-cli** command from a subdirectory under **/opt/SevOne/upgrade**. It can be run from any directory except for from subdirectories under **/opt/SevOne/upgrade**.
For details on **sevone-cli**, please refer to *SevOne Data Insight Troubleshooting / Debug / Tools Guide* > **Tools** > section **SevOne CLI**.

Terminology usage...

In this guide if there is,

- [any reference to *master*] OR
- [[if a CLI command *(*for *NMS* or *Kubernetes* or *Redis)* contains *master*] AND/OR
- [its output contains *master*]],

it means *leader* or *control plane*.

And, if there is any reference to *slave* or *worker*, it means *follower* or *agent*.

Please refer to *SevOne Data Insight User Guide* for information on using Data Insight and the *SevOne Data Insight Installation Guide* for information on installing Data Insight.

2 Relevant Terms

Term	Definition
Authentication Source	The server that the tenant and user you specify authenticate to. This would be the same as the data source (a SevOne NMS appliance or cluster) where SevOne Data Insight gets data from.
Data Insight Instance	An umbrella term that is synonymous with DI cluster.
Data Insight 'control plane' Node	A Virtual Machine or appliance that is designated as a Kubernetes *control plane* node.
Data Insight Node	Any Virtual Machine or appliance in a DI instance.
Data Insight 'agent' Node	A Virtual Machine or appliance that is designated as a Kubernetes *agent* node.
Tenant	An isolated group of users who have access to the data of the tenant they belong to.

3 Create YAML File

Allows you to create a new **/opt/SevOne/chartconfs/di_custom.yaml** file if it does not exist already. This file must only contain the settings that you want to override.

1. Using **ssh**, log into SevOne Data Insight as **sevone**.

```
$ ssh sevone@<SevOne Data Insight IP address or hostname>
```

2. Create /opt/SevOne/chartconfs/di_custom.yaml file.

Create /opt/SevOne/chartconfs/di_custom.yaml file

```
$ touch /opt/SevOne/chartconfs/di_custom.yaml
```

4 Configuration

4.1 Helm Chart

SevOne Data Insight Helm chart deploys the application stack on a **Kubernetes** cluster using the **Helm Package Manager**. Please refer to References below for the relevant links.

The Helm chart is configured with a base set of configuration options that can be overridden as needed.

4.1.1 Configure GraphQL

4.1.1.1 Configure Session Idle Timeout

To allow users to remain logged into SevOne Data Insight for more than an hour, execute the steps below.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

   ```
   graphql:
     env:
       AUTH_TIMEOUT: 3600
   ```

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

   ```
   $ sevone-cli playbook up --tags apps
   ```

4.1.1.2 Enable GraphiQL

To enable GraphiQL, execute the steps below. GraphiQL is an in-browser tool for writing, validating, and testing GraphQL queries.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

   ```
   graphql:
     enableGraphiql: true
   ```

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

```
$ sevone-cli playbook up --tags apps
```

4.1.2 Bypass Report Migration Checks

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

```
graphql:
  env:
    SKIP_REPORT_MIGRATION_DRY_RUN: true
```

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

```
$ sevone-cli playbook up --tags apps
```

4.1.3 Enable Shortcuts

Shortcuts are available by default from **SevOne Data Insight** > under **Configure** in left navigation bar.

Enable Shortcuts to allow you to save resource selection as **shortcuts** to be reused across widgets in SevOne Data Insight without having to create groups in SevOne NMS.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

```
ui:
  env:
    FF_ALIASES: true
```

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

```
$ sevone-cli playbook up --tags apps
```

4.1.4 Enable Report Versioning

Enable report versioning to allow user to view saved versions of a report.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

   ```
   reportVersionsEnabled: true
   ```

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

   ```
   $ sevone-cli playbook up --tags apps
   ```

4.1.5 Disable Datasource Operator

In SevOne Data Insight 3.12 and above, requests to SOA are going to round robin to any peer in SevOne NMS clusters that SevOne Data Insight is connected to. This feature can be disabled in SevOne Data Insight to stop providing HA communicating with SevOne NMS and instead, sent requests to the specifically configured NMS peer.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

2. Disable datasource operator.

   ```
   operator:
     datasource:
       enabled: false
   ```

3. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

   ```
   $ sevone-cli playbook up --tags apps
   ```

4.1.6 Disable MySQL Metrics Server

MySQL ships with a metrics server that fails to start up if your environment has **IPv6** enabled.

⚠ Until SevOne Data Insight supports **IPv4** / **IPv6** dual stack environments, MySQL metrics server must be disabled.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

2. Disable MySQL Metrics Server.

```
mysql:
  metrics:
    enabled: false
```

3. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

```
$ sevone-cli playbook up --tags apps
```

4.1.7 Change Prometheus Password

To change the password for prometheus, execute the steps below.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to change the password using the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

```
prometheus:
  auth:
    username: datainsight
    password: datainsight
```

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

```
$ sevone-cli playbook up --tags apps
```

4.1.8 Cron Schedules

SevOne Data Insight has a number of cron jobs to perform tasks such as API key rotation, syncing users from SevOne NMS to SevOne Data Insight, and sweeping temporary assets from the object store.

The following table contains the default schedules for each job.

Job	Cron Schedule	Description
apikey-rotation	`0 5 */30 * *`	At 05:00 on every 30th day-of-month.
asset-sweeper	`*/5 * * * *`	At every 5th minute.
user-sync	`*/5 * * * *`	At every 5th minute.

Custom schedule can be set via **/opt/SevOne/chartconfs/di_custom.yaml** file.

Examples

for SevOne Data Insight 3.11.x and below

```
crontab:
  apikey-rotation: "0 5 */30 * *"
  asset-sweeper: "*/5 * * * *"
  user-sync: "*/5 * * * *"
```

for SevOne Data Insight 3.12.x and above

```
crontab:
  apikey-rotation:
    schedule: "0 5 */30 * *"
  asset-sweeper:
    schedule: "*/5 * * * *"
  user-sync:
    schedule: "*/5 * * * *"
```

4.1.9 Resource Requests & Limits

for Advanced Users Only

Please refer to **Kubernetes Resource Requests & Limits** link in References section for additional information on configuring resource requests and limits.

Many of the deployments within SevOne Data Insight have resource requests and limits configured by default. Please refer to the example below containing the defaults for each deployment.

These have been generally configured with wide enough parameters to accommodate heavy usage of SevOne Data Insight.

However, should your usage be atypical and you find that a deployment lacks sufficient resources, or perhaps if you find that a deployment's usage is not very high and would like to scale it back to leave more room for other deployments/processes, you can apply custom requests and limits for a deployment in /opt/SevOne/chartconfs/di_limits.yaml file.

Example

Custom Requests & Limits

```
# /opt/SevOne/chartconfs/di_limits.yaml
graphql:
  resources:
    requests:
      cpu: 500m
      memory: 4000Mi
    limits:
      cpu: 2000m
      memory: 4000Mi
mysql:
  primary:
    resources:
      requests:
        cpu: 2000m
        memory: 4000Mi
      limits:
        cpu: 3500m
        memory: 8000Mi
printer:
  resources:
    requests:
      cpu: 500m
      memory: 4000Mi
    limits:
      cpu: 2000m
      memory: 4000Mi
scheduler:
  resources:
    requests:
      cpu: 50m
      memory: 100Mi
    limits:
      cpu: 200m
      memory: 250Mi
ui:
  resources:
    requests:
      cpu: 50m
      memory: 100Mi
    limits:
      cpu: 100m
      memory: 200Mi
prometheus:
  server:
    resources:
      requests:
        cpu: 500m
        memory: 4000Mi
      limits:
        cpu: 1000m
        memory: 8000Mi
nodeExporter:
  resources:
```

```
    requests:
      cpu: 100m
      memory: 100Mi
    limits:
      cpu: 200m
      memory: 100Mi
kubeStateMetrics:
  resources:
    requests:
      cpu: 50m
      memory: 100Mi
    limits:
      cpu: 100m
      memory: 100Mi
rabbitmq:
  resources:
    requests:
      cpu: 400m
      memory: 250Mi
    limits:
      cpu: 800m
      memory: 750Mi
wdkserver:
  resources:
    requests:
      cpu: 50m
      memory: 1000Mi
    limits:
      cpu: 100m
      memory: 2000Mi
redis:
  master:
    resources:
      requests:
        cpu: 100m
        memory: 300Mi
      limits:
        cpu: 250m
        memory: 1000Mi
minio:
  resources:
    requests:
      cpu: 100m
      memory: 2500Mi
    limits:
      cpu: 250m
      memory: 4000Mi
cron:
  resources:
    requests:
      cpu: 750m
      memory: 500Mi
    limits:
      cpu: 1000m
      memory: 1000Mi
```

Disable Limits

```
# /opt/SevOne/chartconfs/di_limits.yaml
graphql:
  resources: null

mysql:
  primary:
```

```
    resources: null

printer:
  resources: null

scheduler:
  resources: null

ui:
  resources: null

prometheus:
  server:
    resources: null
  nodeExporter:
    resources: null
  kubeStateMetrics:
    resources: null

rabbitmq:
  resources: null

wdkserver:
  resources: null

redis:
  master:
    resources: null

minio:
  resources: null

cron:
  resources: null
```

4.1.10 Configure HELM

This applies for Advanced Users Only.

By default, SevOne Data Insight does not ship with **helm** configured so that users can apply the changes via **sevone-cli**. This ensures that any misconfigurations or issues with the deployment will be automatically rolled back.

This can take some time to complete since **helm** waits for all pods to successfully start up before finishing. If you are an advanced user and know what you are doing, you may apply the changes directly using **helm**.

1. To configure **helm**, execute the following steps.

```
$ helm repo add --force-update --insecure-skip-tls-verify \
local https://localhost:6443/static/charts

$ helm repo update
```

2. Apply the changes directly via **helm**.

```
$ helm upgrade di local/di --insecure-skip-tls-verify \
```

```
--install --devel -f /opt/SevOne/chartconfs/di_custom.yaml
```

4.1.11 Configure Maps

IMPORTANT: Missing / misconfigured tile server for Map mode

The tile server for Map (geospatial) mode is missing or misconfigured in SevOne Data Insight 6.6.0.
To resolve this, you are **required** to perform additional configuration to obtain the proper tiles for the maps.

- if **using Command Line Interface**,
  - you are **required** to execute the commands below to configure variables **tileserver** and **attribution** under **maps** and apply the changes.
  - you **must** perform an upgrade to SevOne Data Insight 6.6.1 or above. For details on how to upgrade using the command line interface, please refer to *SevOne Data Insight Upgrade Process Guide* > section **using Command Line Interface**.
- if **using Graphical User Interface**,
  - you **must** perform an upgrade to SevOne Data Insight 6.6.1 or above. Please refer to *SevOne Data Insight Upgrade Process Guide* > section **using Graphical User Interface** > at step **Configure**, configure variables **Tileserver** and **Attribution** under **Maps** and successfully complete the upgrade.

NOTICE

Failure to set variable *maps.tileserver* while performing an upgrade to SevOne Data Insight 6.6.1 or above, will convert your current map in **Maps** mode to **Canvas** mode without the ability to switch back to Map mode until the tile server is provided.
If the owner of the selected tile server requires to be given credit for the tiles being used, set variable *maps.attribution*. Text entered in variable *maps.attribution* will appear in the bottom right corner of each map.

For further assistance, please contact **SevOne Support Team** or your **Technical Account Manager** to configure this properly.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

2. Configure the *maps* variable.

   To create the tile server URL with the Access Token, please refer to section Create Tile Server URL below.

```
maps:
  tileserver: "<enter tileserver URL containing the token>"
  attribution: "<enter attribution for a given tileserver if required by the tile's owner>"
```

Example: for Dark mode

```
maps:
  tileserver: "https://api.mapbox.com/styles/v1/sevone/cl1fjv99c01mp01ql9neoau07/tiles/256/
{z}/{x}/{y}@2x?access_token=<ACCESS_TOKEN>"
  attribution: "Some attribution; props to stamen for providing tiles, etc."
```

Example: for Light mode

```
maps:
  tileserver: "https://api.mapbox.com/styles/v1/sevone/cl1fl73f101qb01qnc7wh22i5/tiles/256/
{z}/{x}/{y}@2x?access_token=<ACCESS_TOKEN>"
  attribution: "Some attribution; props to stamen for providing tiles, etc."
```

3. Apply the change made to /opt/SevOne/chartconfs/di_custom.yaml file.

```
$ sevone-cli playbook up --tags apps
```

4.1.11.1 Create Tile Server URL

To create the Access Token, use may use mapbox (*recommended*) or any other tool of your preference. The steps below show how to create the Access Token using *mapbox* - this is only an example.

1. Create a *mapbox* account. Enter the following URL in a web browser of your choice.

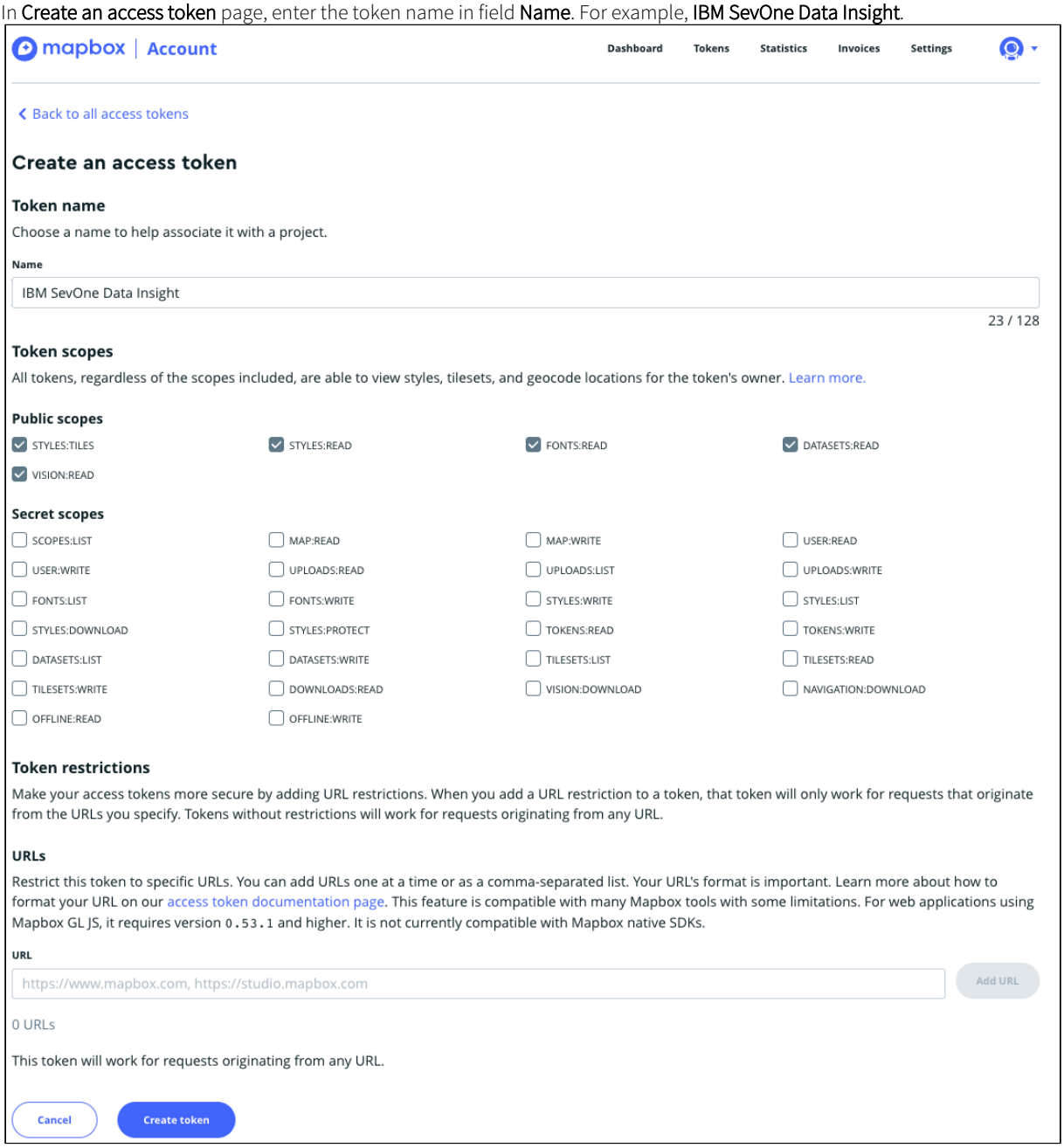
```
https://mapbox.com
```

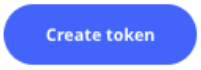
⚠ You only need to create your mapbox account once. Please save your username / email address and password for future use.

2. Once the *mapbox* account is created, sign in.
3. Click 

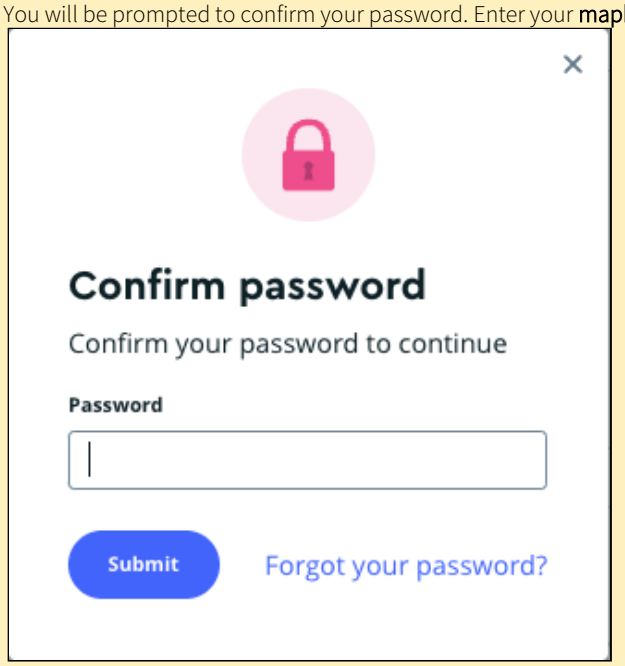
button.

4. In **Create an access token** page, enter the token name in field **Name**. For example, **IBM SevOne Data Insight**.



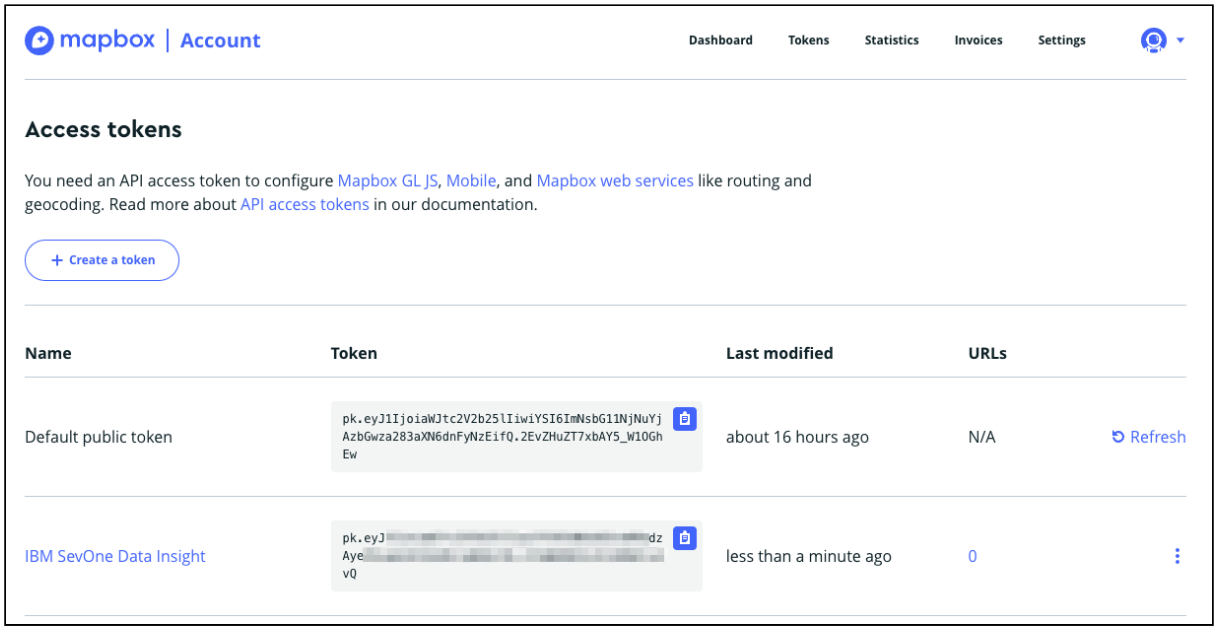
5. Click  button to create the mapbox Access Token.

- You will be prompted to confirm your password. Enter your **mapbox** account password.



- Click **Submit**.

6. Your Access Token for token name *IBM SevOne Data Insight* is created. For security purposes, the token has been obscured in the screenshot below.



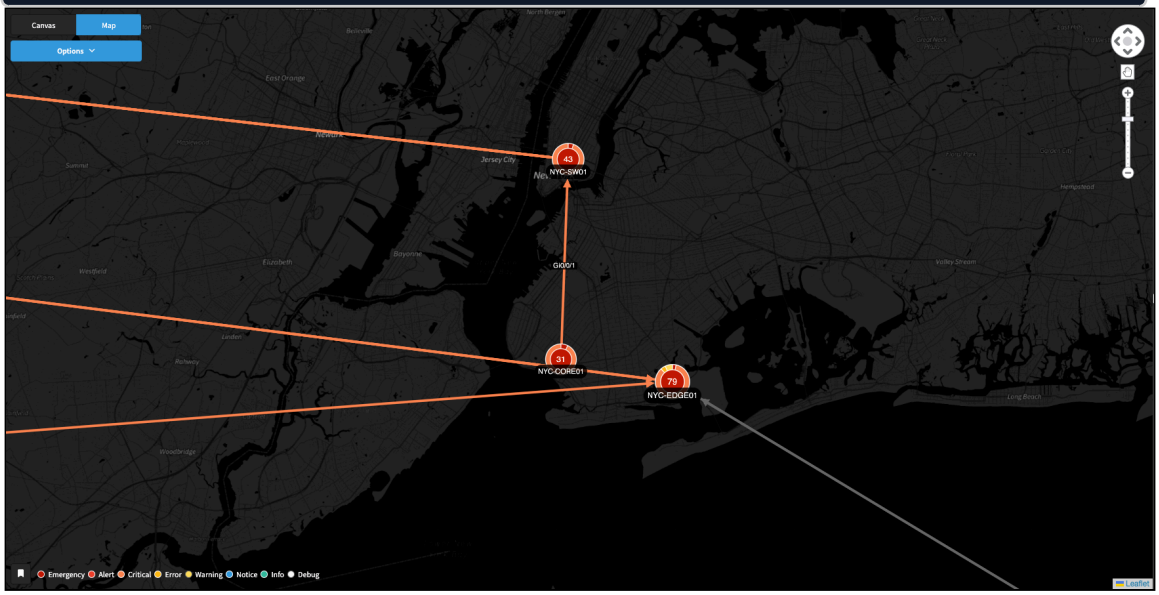
7. For SevOne Data Insight, you have *dark* and *light* modes. Based on the mode, your tile server URL is created.

for Dark mode,

In the URL, replace **<ACCESS_TOKEN>** with the Access Token generated above for your token name, **IBM SevOne Data Insight**.

Tile Server URL for Dark mode

```
https://api.mapbox.com/styles/v1/sevone/cllfjv99c01mp01ql9neoau07/tiles/256/{z}/{x}/{y}@2x?
access_token=<ACCESS_TOKEN>
```

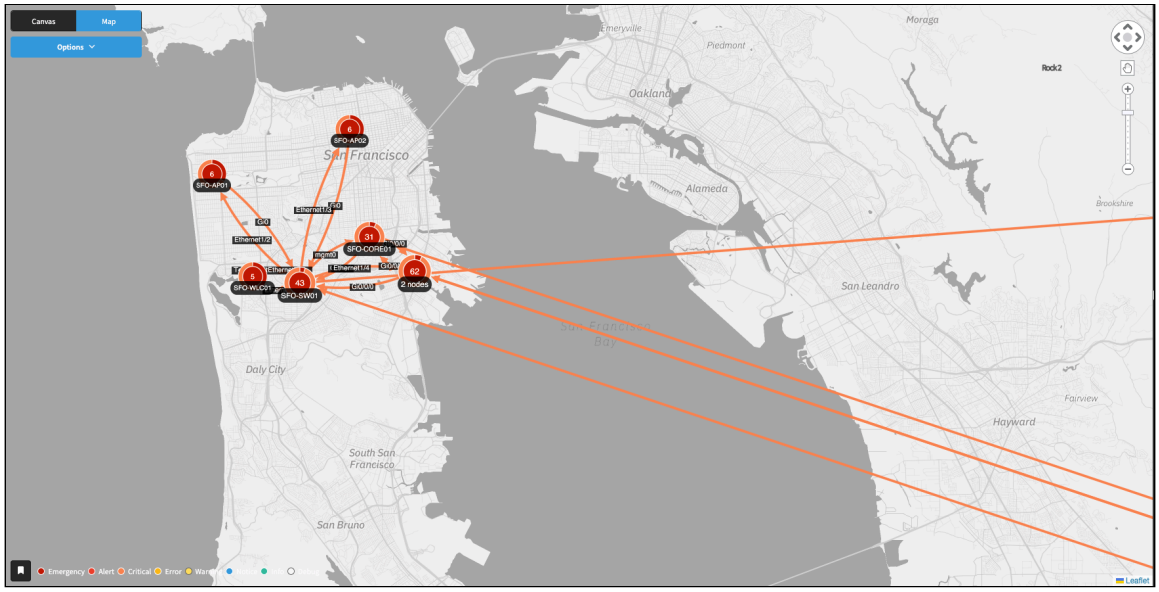


for Light mode,

In the URL, replace **<ACCESS_TOKEN>** with the Access Token generated above for your token name, **IBM SevOne Data Insight**.

Tile Server URL for Light mode

```
https://api.mapbox.com/styles/v1/sevone/cllfl73f101qb01qnc7wh22i5/tiles/256/{z}/{x}/{y}@2x?
access_token=<ACCESS_TOKEN>
```



4.2 Cloud Builds

4.2.1 Amazon Web Services

If you are using an AMI build of SevOne Data Insight and would like to install **AWS CloudWatch** and **AWS SSM** agents, **sevone-cli** provides a convenient subcommand.

```
$ sevone-cli cloud-setup
```

After both AWS agents are installed, you can proceed to configure them as outlined on the AWS documentation website. For details on **AWS CloudWatch**, please refer to the relevant link in References.

4.3 NGINX Headers

For a variety of reasons you may want to configure **nginx** to set certain headers when using SevOne Data Insight. This can also be used to nullify large headers.

4.3.1 Create kubeobjects Directory

1. Create **kubeobjects** directory.

```
$ mkdir /opt/SevOne/kubeobjects
```

4.3.2 Create custom-headers.yaml

1. Determine the headers you want to change and create a file, **custom-headers.yaml** file.

Create /opt/SevOne/chartconfs/di_custom.yaml file

```
$ touch /opt/SevOne/kubeobjects/custom-headers.yaml
```

2. Add the following contents with your custom headers set under the **data** section.

Create /opt/SevOne/kubeobjects/custom-headers.yaml file

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: custom-headers
  namespace: default
data:
  X-Different-Name: "true"
  X-Request-Start: t=${msec}
  X-Using-Nginx-Controller: "true"
```

4.3.3 Update Ingress Configuration

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/ingress_custom.yaml** file to set environment variable **controller** and then, save it.

⚠ If **/opt/SevOne/chartconfs/ingress_custom.yaml** file does not exist, please *create* it.

```
$ touch /opt/SevOne/chartconfs/ingress_custom.yaml
```

```
controller:
  config:
    proxy-set-headers: "default/custom-headers"
```

4.3.4 Update Ansible Inventory

1. Using a text editor of your choice, edit **/etc/ansible/group_vars/all/custom-headers.yaml** file to set environment variable **kube_object_ingress** and then, save it.

⚠ If **/etc/ansible/group_vars/all/custom-headers.yaml** file does not exist, please *create* it.

```
$ touch /etc/ansible/group_vars/all/custom-headers.yaml
```

```
kube_object_ingress:
  enabled: true
  files:
  - /opt/SevOne/kubeobjects/custom-headers.yaml
```

4.3.5 Apply your Changes

```
$ sevone-cli cluster up
```

4.3.6 Force NGINX Config Reload

When checking the **nginx** config, you may find that the custom headers configmap and the configuration differ. This typically happens based on which files are loaded when bringing up SevOne Data Insight.

1. To force the **nginx** config to reload, change the ingress controller config map.

```
$ kubectl edit cm ingress-ingress-nginx-controller
```

This will open the NGINX config file in text editor, **vi**. Find the **data** section and change variable **keep-alive-requests**.

```
data:
  keep-alive-requests: "10001"
```

2. To save and exit the file, enter **:wq!**
3. Kubernetes will automatically regenerate NGINX's config.

4.4 OOTB Reports

OOTB reports and templates can be imported using **sevone-cli**.

> ⚠ You will be prompted for your login credentials.
> **admin** is the default username. If you would like the username to be something other than *admin*, pass the **-u** option in the command.

```
$ sevone-cli sdi reports load /opt/SevOne/upgrade/utilities/ootb-reports/sdi-v*.tar
```

To import the reports to a specific tenant, or if your tenant name is not the default *SevOne*, pass in the **--tenant** or **-t** flag.

Example

```
$ sevone-cli sdi reports load /opt/SevOne/upgrade/utilities/ootb-reports/sdi-v*.tar \
--tenant "<enter tenant name>"
```

4.5 PDF Printing

4.5.1 Configuration

4.5.1.1 Environment Variables

The printer pod is configured by environment variables. Normally, no manual configuration is required. With administrator discretion, the following environment variables are available.

Name	Default	Description
PRINTER_DEBUG	false	Set **true** to enable debug logging.
PRINT_REPORT_CONCURRENCY	5	Maximum concurrent print requests.
PRINT_WIDGET_DPI	300	Widget image resolution.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

   Enable debug logging

   ```
   printer:
     env:
       PRINTER_DEBUG: "true"
   ```

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

   ```
   $ sevone-cli playbook up --tags apps
   ```

4.5.1.2 Debug Logging

When enabled, additional information is logged.

- Browser console messages.

- HTTP requests and responses like the Network tab in dev tools.
- Report structure info, such as sections and widgets being printed.
- Event communication between the printer and User Interface.

4.5.1.3 Report Concurrency

It is **not** advised to increase this setting. Though, it might make sense to decrease it in specific situations if the Page Crashed Error is occurring on certain reports.

Increase report concurrency by scaling the printer pod, instead.

4.5.1.4 Scale Printer Pod

The printer pod can be scaled up to the same number of *agent* nodes in your cluster. Each pod replica can process an additional 5 concurrent requests.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

   Set desired number of printer replicas

   ```
   printer:
     replicas: 3
   ```

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

   ```
   $ sevone-cli playbook up --tags apps
   ```

4.5.1.5 Widget Resolution

Default resolution is high enough to produce smooth graphics for display and printing on high resolution devices.

If PDF file size becomes a problem, this setting can be used to decrease file size by reducing the resolution.

4.5.2 Troubleshooting

The printer pod is a service that processes all print requests, generating a PDF document, and storing it in **minio**. It works by running a headless Chromium browser to SevOne Data Insight User Interface. It generates a per-widget view and screen captures each widget to an image. These images are compiled into a formatted PDF document.

The print routine is CPU and memory intensive. Therefore, it was designed to be scalable. By default, it can process 5 concurrent print requests. Additional requests are queued up.

4.5.2.1 Timeout

The printer allows up to 15 minutes to print a report before timing out. If a report contains many widgets that take a significant amount of time to render, it should be split up into smaller reports.

If there is a problem during print, it is possible the report will timeout early at 30 seconds because it was unable to start the User Interface in the browser. This appears in the log as:

```
Timeout waiting for reportload
```

4.5.2.2 Page Crashed Error

The Chromium browser can sometimes report a **Page Crashed Error** in the printer logs. This indicates an *out of memory* condition that caused the browser to fail to render.

Verify the Kubernetes host has available memory for the pod. Check for any printer configuration in **/opt/SevOne/chartconfs/di_custom.yaml** that might increase resource utilization. For example, environment variable **`PRINT_REPORT_CONCURRENCY`** can impact memory utilization.

4.5.2.3 RabbitMQ Queues

The printer uses a queue in **RabbitMQ** to manage incoming requests. This allows requests to be captured even when the printer is unavailable to handle them at that moment.

Queue	Description
printReport	Print reports (on-demand or scheduled).
printerHealth	Internal health check.
printerScheduledReport	Triggered scheduled prints.
schedulerHealth	Internal health check.
schedulerMutation	Internal change requests for scheduled reports.
schedulerQuery	Internal query requests for scheduled reports.

4.5.2.4 Check Queues

If the printer is overloaded, the queue will grow in size and you will notice print requests are not completing. This can be monitored by opening a terminal to the **rabbitmq** pod.

In this example,

The **printReport** queue contains `2` queued up requests. This means the printer is busy and the `2` requests are waiting to be processed.
You may find additional **amq.gen-xxx** queues. These are the temporary queues generated by clients accessing these queues. These should disappear after their request is processed.

Example

```
$ kubectl exec -it sts/di-rabbitmq -- rabbitmqctl list_queues
Timeout: 60.0 seconds ...
Listing queues for vhost / ...
name    messages
schedulerMutation   0
printScheduledReport    0
printReport 2
printerHealth   0
schedulerHealth 0
schedulerQuery  0
```

4.5.2.5 Clear a Queue

If there is a problem and you need to clear the requests, execute the following command. Log into SevOne Data Insight as **sevone** using **ssh**.

```
$ sevone-cli exec di-rabbitmq -- rabbitmqctl purge_queue printReport
```

4.5.2.6 Scheduled Report Caching

SevOne Data Insight maintains a cache of the printed PDF files for scheduled reports. Depending on your usage of report scheduling, it is recommended to occasionally clean up the cache to free up disk space.

Execute the following command to delete files older than one week (604800 seconds).

```
$ sevone-cli exec graphql -- "npm run asset-sweeper -- \
--prefix=scheduledReports --age=604800"
```

4.6 OpenID Connect

SevOne Data Insight may be configured to use OpenID Connect as an authentication mechanism.

4.6.1 Enable OpenID Connect

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file, add section **oidc**, and save the file.

   ⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

   Add section 'oidc'

```
## OpenId Connect authentication support
oidc:
  enable: true
  enableRedirect: true
  authority: "https://auth.example.com/sso"
  clientId: "datainsight"
  clientSecret: "secret"
  responseType: "code"
  scope: "openid profile groups"
  groupPrefix: "tenant:"
```

⚠ Update the property values with those specific to **OpenID Connect (OIDC)** authentication source.

Property Name	Value Type	Example	Description
enable	boolean	true	Whether to enable OIDC.
enableRedirect	boolean	true	Whether to redirect when not authenticated (usually true).
authority	string URL	"https://auth.example.com/sso"	Server's authorization endpoint URL defined by RFC 6749. For example, "https://<SevOne NMS appliance>/sso".
clientId	string	"datainsight"	The client identifier of the relying party at the provider.
clientSecret	string	"s2gR2wGwvnF4rikxnwxR"	Client secret of the relying party at the provider.
responseType	string	"id_token"	Type of response. Either "code" or "id_token".
scope	string	"openid profile groups"	Requested OAuth2 scopes. "groups" is only for multi-tenant.
groupPrefix	string	"sevonetenant:"	A group claim prefixed with this determines the tenant.

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

```
$ sevone-cli playbook up --tags apps
```

4.6.2 The groupPrefix Property

The properties listed above are standard OpenID Connect configuration values, with the exception of **groupPrefix**. The *groupPrefix* property is used to help determine which tenant is used for login. If the provider provides a *groups* claim (as an array of strings), and the *groupPrefix* is defined, then the user will be logged in using the first value in the claim that has this prefix (with the prefix removed).

Example

A groups claim of **["tenant:Test1", "example2"]** will login with the tenant **Test1**, if the *groupPrefix* property is set to **tenant:**.

⚠ The groups claim is provided by the Single Sign-On provider, for SevOne Data Insight to work on a Single Sign-On setup with multiple tenants, there must be a groups claim that indicates which tenant the user belongs to.

4.6.3 Configure OIDC using SevOne NMS as a provider

1. Ensure the SevOne NMS **dex** setup is working. Please refer to *SevOne SAML Single Sign-On Setup Guide* for details.
2. Create a new *static client* for SevOne Data Insight with a unique ID. This ID will be used as the **clientId** when configuring SevOne Data Insight.
3. Ensure the **redirectURIs** for SevOne Data Insight static client point to **https://<IP address or hostname for SevOne Data Insight>/callback**.

Example: Configuration

```
staticClients:
- id: sevonedatainsight
  redirectURIs:
  - 'https://10.10.10.10/callback'
  name: 'SevOne DI'
  secret:
Uhy6g7CyTALSIrwsnEYNBF60ZM0eeHQkZCqIoXBrPOoiZjJANcNZb7CJGB8wGDrE1sGbM9pdIzSJ0OL0PfyNXeg9Uih7F9
FNdTaVyxMw8evFtcq4L7z71IuBJ08V4rRH
  samlInitiated:
    redirectURI: https://10.10.10.10/callback
```

4.6.4 OIDC with Multiple Datasources

As with all SevOne Data Insight tenants that have multiple datasources, the username extracted from the OIDC provider must have a user present on every NMS Datasource that is part of the tenant.

4.6.5 OIDC with non-standard Login Claims

1. If the Single Sign-On provider does not have the user's login stored under **name**, using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** to configure environment variable **OIDC_NAME_CLAIM**, to allow SevOne Data Insight to use another field in the JWT to determine the user's login name.

⚠ If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

Example: Configuration

```
graphql:
  env:
```

```
    OIDC_NAME_CLAIM: email
```

Save /opt/SevOne/chartconfs/di_custom.yaml file.

2. Apply the change made to /opt/SevOne/chartconfs/di_custom.yaml file.

```
$ sevone-cli playbook up --tags apps
```

4.7 Tenants and Datasources

4.7.1 Manage Tenants

4.7.1.1 Create a Tenant

To add a tenant, execute the following command. The prompts are similar to Connect NMS in *SevOne Data Insight Installation Guide* > section Install Single-Node.

```
$ sevone-cli exec graphql -- npm run create-tenant
```

4.7.1.2 Modify a Tenant

```
$ sevone-cli exec graphql -- npm run reconfig-tenant
```

4.7.2 Manage Datasources via CLI

When prompted to Login instead of providing an API key you should type y.
This will generate the API key for you based off of the NMS username and password you provide.

4.7.2.1 Create a Datasource

1. SOA is a required dependency and must be installed on your SevOne NMS.

You will be prompted to enter a password. This will be the root password for your NMS.

SOA version

SOA must be on the latest version on all appliances in SevOne NMS cluster. Command Line Interface (CLI) must be used to upgrade SOA on all peers as the graphical user interface (GUI) only upgrades SOA for the NMS appliance you are connected to.

⚠ Add flag **--all-peers** if you want to install / upgrade SOA on all peers in the cluster.

```
$ sevone-cli soa upgrade \
/opt/SevOne/upgrade/utilities/SevOne-soa-*.rpm \
<enter SevOne NMS IP address> --all-peers
```

2. Create the datasource.

```
$ sevone-cli exec graphql -- npm run create-datasource
```

4.7.2.2 Modify a Datasource

If you need to modify the datasource information such as, IP address or SOA API Key, please execute the command below.

⚠ To maintain the existing settings of the datasource, leave the prompt **empty** and press **<ENTER>**.

```
$ sevone-cli exec graphql -- npm run reconfig-datasource
```

4.7.3 Manage Datasources via User Interface

4.7.3.1 Obtain Authentication Token

⚠ As of SevOne NPM release 6.6, authentication token is no longer required.

1. Using a web browser of your choice, enter the URL for SevOne NMS appliance you want to connect SevOne Data Insight to.

```
https://<enter SevOne NMS hostname or IP address>/api/v3/docs
```

2. Choose REST API version as **Version 3**.
3. Click on **Users** to view Users operations.
4. Under **Users**, click on **POST** endpoint **/api/v3/users/signin**.
5. Under **Parameters**, all the way to the right, locate the **Model Schema** field. Click on the field to copy its content under **Value**.
6. On the left side of the **Parameters** section, locate the body under Value. After **"username":**, replace **string** with the corresponding SevOne NMS user name. Make sure to enter it within the quotes.
7. After **"password":**, replace **string** with the corresponding SevOne NMS password. Make sure to enter it within the quotes.
8. At the bottom of the **POST** section, click the **Try it out!** button.

9. Scroll down to the **Response Body** field. You should see a long alphanumeric string after **<token>**. This is the token that you need. Double-click the token to select it. Then copy it.
10. In the upper right corner of the **SevOne API Documentation** page, locate the **bearer token** field. Paste the token into this field. You should now have permissions to perform operations.
11. Now, *obtain the API Key*.

4.7.3.2 Obtain API Key

1. Under Users, click on **POST** endpoint **/api/v3/users/apikey**.
2. Under **Parameters**, all the way to the right, locate the **Model Schema** field. Click on the field to copy its content under **Value**.
3. On the left side of the **Parameters** section, locate the body under Value. After **"application":**, replace **string** with a unique identifier. For example, *testAPIKey*. Make sure to enter it within the quotes.
4. At the bottom of the **POST** section, click the **Try it out!** button.
5. Scroll down to the **Response Body** field. You should see a long alphanumeric string after **<apiKey>**. This is the API Key that you need. Double-click the key to select it. Then copy it.
6. You are now ready to create a datasource.

4.7.3.3 Create a Datasource

1. Using a web browser of your choice, enter the URL for your SevOne Data Insight.
2. From the left navigation bar, click **Configure** and select **Data Sources**. You are now on **Datasource Manager** page.
3. From the drop-down, select **NMS**, for example.
4. In the **Name** field, enter the name of the datasource you want to create. For example, *myDatasource*.
5. In **API Server** field, enter the URL for SevOne NMS appliance that is connected to this SevOne Data Insight.
6. By default, **Authentication** field is set to **Token**.
7. In **API Token** field, paste the API Key obtained in step 5. under section *Obtain API Key*.
8. From **Type** drop-down, choose **METRICS/FLOW** option.
9. Click **Test connection** button to confirm you can connect to the server.
10. Click **Add Datasource** in the upper-right corner. You will see your datasource, *myDatasource*, available from the **ACTIVE DATASOURCES** list.

4.7.3.4 Modify a Datasource

1. From **ACTIVE DATASOURCES** list, select the datasource to view its details or to modify its configuration.
2. After modifying, click **Save**.

> As you modify the fields, real-time health checks are performed.

4.8 TLS Certificates

4.8.1 New or Rotating Certificates

> If you are renewing the TLS certificates, you must first delete the existing Kubernetes *secret* before continuing.
>
> ```
> $ kubectl delete secret datainsightcerts
> ```

If you have TLS certificates, you may load them into Kubernetes as a *secret* and configure SevOne Data Insight to utilize them.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to enable TLS certificates and then, save it.

> If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

Enable the certificates

> **IMPORTANT**
>
> Please make sure to copy and paste the configuration below as-is in **/opt/SevOne/chartconfs/di_custom.yaml** file. Spacing is very important and it must match when copied into */opt/SevOne/chartconfs/di_custom.yaml* file.

```
ingress:
  hosts:
  - "di.customer.com"
  tls:
  - secretName: datainsightcerts
    hosts:
    - "di.customer.com"
```

2. Load your TLS certificates as a Kubernetes secret.

```
$ ls /home/sevone/tls
tls.crt  tls.key

$ kubectl create secret tls datainsightcerts \
--key /home/sevone/tls/tls.key \
--cert /home/sevone/tls/tls.crt
```

3. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

```
$ sevone-cli playbook up --tags apps
```

4.8.2 Fix an Invalid TLS Secret Name

SevOne Data Insight expects the TLS certificate secret to be specifically named **datainsightcerts**.

If you find that your secret is named something else, please rename it to the expected name to avoid any issues.

4.8.2.1 Find your Config File

SevOne Data Insight Version	Config File Location
SDI 3.4.x and below	/opt/datainsight/values-override.yaml
SDI 3.5.x and above	/opt/SevOne/chartconfs/di_custom.yaml

4.8.2.2 Find your Secret Name

1. In the config file, TLS secret name will be present in the location <**SECRET_NAME**>.

> **IMPORTANT**
>
> Please make sure to copy and paste the configuration below as-is in the configuration file location (for SDI 3.4.x and below, in **/opt/datainsight/values-override.yaml** file and for SDI 3.5.x and above, in **/opt/SevOne/chartconfs/di_custom.yaml** file). Spacing is very important and it must match when copied into the appropriate configuration file.

```
ingress:
  hosts:
  - "di.customer.com"
  tls:
  - secretName: <SECRET_NAME>
    hosts:
    - "di.customer.com"
```

If **<SECRET_NAME>** is **datainsightcerts**, you are all set and you do not need to perform any other steps. If it does not say **datainsightcerts**, proceed to the next section, Rename your Secret.

4.8.2.3 Rename your Secret

1. Copy the existing certificates to a file.

Example

```
$ kubectl get secret <SECRET_NAME> -o yaml > datainsightcerts.yaml
```

2. Replace all instances of **<SECRET_NAME>** with **datainsightcerts**.

Example

```
$ sed -i 's/<SECRET_NAME>/datainsightcerts/g' datainsightcerts.yaml
```

3. Add your new secret to Kubernetes.

Example

```
$ kubectl apply -f datainsightcerts.yaml
```

4.9 Widget Development Kit

If you want to develop a widget in your environment, you must enable SevOne Data Insight's WDK module.

This module is a collection of docker images that contain the WDK documentation along with the private **npm** registry to install **insight-wdk-cli** on your local machine.

When you enable WDK, SevOne Data Insight exposes two new endpoints.

Endpoint	Description
https://sdi.customer.com/registry	The private **npm** registry.
https://sdi.customer.com/docs	Documentation and HOWTOs on widget development.

widget.json file

widget.json file contains the widget metadata. It is placed in the **root** directory of your widget.

Required Fields

- name
- version
- title
- runtime

Optional Fields

- description
- tags
- exportTypes
- exportMimeTypes
- producedFacets
- consumedFacets

For details on the required and optional fields, please refer to section widget.json.

4.9.1 Enable WDK Module

Enable WDK to expose a private registry and related documentation.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

   If **/opt/SevOne/chartconfs/di_custom.yaml** file does not exist, please create one and add the following to it.

   ```
   wdk:
     enabled: true
   ```

2. Apply the change made to **/opt/SevOne/chartconfs/di_custom.yaml** file.

   ```
   $ sevone-cli playbook up --tags apps
   ```

4.9.2 Configure your Development Environment

On your local machine, configure the following before installing **insight-wdk-cli**.

1. Install **Node.js** if you have not done so already. The recommended version is **v10.24.1**.
2. Install **yarn**.

```
$ npm install -g yarn
```

3. Execute the following commands to configure **npm** and **yarn**.

```
$ npm config set registry https://sdi.customer.com/registry

$ npm config set strict-ssl false

$ yarn config set strict-ssl false
```

4.9.3 Install insight-wdk-cli

On your local machine, you can use **npm** with the **--registry** flag pointing to your SevOne Data Insight instance.

```
$ npm cache clean --force

$ npm i @sevone/insight-wdk-cli@3.0.0-beta.31 -g \
--registry https://sdi.customer.com/registry
```

4.9.4 Migrate to WDK v3.x

Several WDK APIs were deprecated with the release of SevOne Data Insight 3.0.x but were kept for backward compatibility.

As of **insight-wdk@2.3.0**, released with SevOne Data Insight 3.12.x, these APIs will be **disabled by default**.

In an upcoming version, they will be removed completely. To ensure your custom widgets continue to work, these APIs must be replaced with their updated equivalent.

For a detailed migration guide, please refer to **/docs/insight-wdk/v3-migration** endpoint on your SevOne Data Insight instance.

```
https://sdi.customer.com/docs/insight-wdk/v3-migration
```

4.9.4.1 Enable WDK v2.x Compatibility Mode

In case there are custom widgets relying on deprecated APIs, a compatibility mode can be enabled as a **temporary** workaround.

> This mode is a temporary stopgap and should not be relied on as a long term solution.
> It will be removed in an upcoming SevOne Data Insight version. To ensure custom widgets continue working, please make sure you update them as described in the migration guide in the WDK docs.

1. Using a text editor of your choice, edit **/opt/SevOne/chartconfs/di_custom.yaml** file to set the following environment variable and then, save it.

⚠ If /opt/SevOne/chartconfs/di_custom.yaml file does not exist, please create one and add the following to it.

```
ui:
  env:
    WDK2_COMPATIBILITY: true
```

2. Apply the change made to /opt/SevOne/chartconfs/di_custom.yaml file.

```
$ sevone-cli playbook up --tags apps
```

4.9.5 widget.json

widget.json contains the widget's metadata and is placed in the root directory of your widget.

4.9.5.1 Required Fields

4.9.5.1.1 Field '*name*'

- Type: *string*
- Description: The *name* field is a unique identifier for the widget. Here are the rules.
  - name field must be unique
  - name field cannot start with a dot or an underscore
  - uppercase letters are not allowed
  - the name ends up being part of a URL, an argument on the command line, and a folder name. Name cannot contain any non-URL-safe characters.

Example: widget.json, field 'name'

```
{
  "name": "my-widget"
}
```

4.9.5.1.2 Field '*version*'

- Type: *string*
- Description: This contains the widget's *version* and it must be parseable by *node-semver*. Changes to the widget come along with the changes to the version when published.

Example: widget.json, field 'version'

```
{
  "version": "3.5.0"
}
```

4.9.5.1.3 Field '*title*'

- Type: *string*
- Description*:* the *title* is a user-friendly display name for the widget.

Example: widget.json, field 'title'

```
{
  "title": "My Widget"
}
```

4.9.5.1.4 Field '*runtime*'

- Type: *string*
- Description*: runtime* represents which version of WDK is being used to develop against. This field is automatically generated and you will not need to modify this field.

Example: widget.json, field 'runtime'

```
{
  "runtime": "3.0.0"
}
```

4.9.5.2 Optional Fields

4.9.5.2.1 Field '*description*'

- Type: *string*
- Description*:* The *description* field contains the description of the widget. It helps user to discover the widget.

Example: widget.json, field 'description'

```
{
  "description": "Historical temperature data for a city."
}
```

4.9.5.2.2 Field '*tags*'

- Type: Array<string>
- Description: *tags* is a set of keywords to describe the widget. It is useful for widget discovery.

Example: widget.json, field 'tags'

```
{
  "tags": [ "weather", "location", "time series" ]
}
```

4.9.5.2.3 Field '*exportTypes*'

- Type:

```
{
  [mimeType: string]: Array<ExportTypeConfigType>
}
```

```
type ExportTypeConfigType = {
  value: string,
  label: string
};
```

- Description: An object which maps from MIME type keys to an array of export types for each MIME type. This field was introduced in SevOne Data Insight 3.12 or *insight-widget-runtime* 1.3.0-beta.12. To support older versions, you do not need to also pass *exportMimeTypes*. However, please refer to the note about including a fallback in the widget's exportData function.

Example: widget.json, field 'exportTypes'

```
{
  "exportTypes": {
    "text/csv": [
      {
        "value": "AS_VISUALIZED",
        "label": "As Visualized"
      },
      {
        "value": "TIME_SERIES",
        "label": "Time Series"
      },
      {
        "value": "SUMMARY",
        "label": "Summary"
      }
    ],
```

```
    "application/json": [
      {
        "value": "AS_VISUALIZED",
        "label": "As Visualized"
      },
      {
        "value": "RAW_DATA",
        "label": "Raw Data"
      }
    ]
  }
}
```

For simpler use-cases where only one export type is needed for each MIME type, you may use field *exportMimeTypes*.

4.9.5.2.4 Field '*exportMimeTypes*'

- Type: Array<string>
- Description: This is a set of MIME types representing the formats the widget can export data as.

Example: widget.json, field 'exportMimeTypes'

```
{
  "exportMimeTypes": [ "text/csv", "application/json" ]
}
```

Field *exportTypes* takes precedence over field *exportMimeTypes* if both fields are passed. Field *exportMimeTypes* is still supported by itself for simpler use-cases where multiple export types per MIME type are not needed.

4.9.5.2.5 Field '*producedFacets*'

- Type: Array<FacetSchema>
- Description: This is a set of facet schemas the widget may broadcast.

Example: widget.json, field 'producedFacets'

```
{
  "producedFacets": [
    {
      "$id": "my-location",
      "type": "object",
      "properties": {
        "city": {
          "type": "string"
        },
        "state": {
          "type": "string"
        }
      },
      "required": [ "city", "state" ]
    }
```

```
  ]
}
```

4.9.5.2.6 Field 'consumedFacets'

- Type: Array<FacetSchema>
- Description: This is a set of facet schemas the widget is capable of handling and processing internally into the configuration.

Example: widget.json, field 'consumedFacets'

```
{
  "consumedFacets": [
    {
      "$id": "my-location",
      "type": "object",
      "properties": {
        "city": {
          "type": "string"
        },
        "state": {
          "type": "string"
        }
      },
      "required": [ "city", "state" ]
    }
  ]
}
```

Function exportData()

```
type ExportTypeConfigType = {
  value: string,
  label: string
};

(
  configuration: Config,
  data: Data,
  mimeType: string,
  exportTypeConfig?: ExportTypeConfigType
) => Blob | Promise<Blob | null> | null
```

A widget can be asked to export its data as any valid MIME type it has registered in its *widget.json* file. This data must be represented as a **Blob**. If **null** is returned, no further action will be taken.
The *exportTypeConfig* argument includes further type information for widgets that can export multiple export types of the same mime type. This argument will only be available if the *widget.json* file uses exportTypes rather than exportMimeTypes. This is available in *insight-widget-runtime* version **1.3.0-beta.12 or later**, which is shipped with SevOne Data Insight 3.12 or later.
To support older versions of the runtime or SevOne Data Insight, this argument should always be considered *optional*, and the export function should include a default fallback in case *mimeType* is passed, but *exportTypeConfig* is not.

Mime Types

Please refer to https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types for the **MIME types (IANA media types)** available.

Facet Schemas

A facet schema is simply a valid JSON schema with a required **$id** field.

```
{
  "$id": "my-timespan",
  "type": "object",
  "properties": {
    "startTime": {
      "type": "number",
      "description": "Start time."
    },
    "endTime": {
      "type": "number",
      "description": "End time."
    },
    "timezone": {
      "type": "string"
    }
  },
  "required": [ "startTime", "endTime" ]
}
```

4.10 SevOne NMS to SevOne Data Insight Drillback URL Configuration Tutorial

The drillback URL configuration tutorial can also be found in *SevOne NMS User Guide* > section *Webhook Definition Manager*.

IMPORTANT

NMS cluster must be on SevOne NMS *6.3.1 or higher*.

This section explains the SevOne Data Insight URL structure, how to create the structure from SevOne NMS > **Events** > **Configuration** > **Webhook Definition Manager**, the origin of each of its constituent URL components, and the prerequisites required in SevOne Data Insight and SevOne NMS for this integration to work.

4.10.1 SevOne Data Insight Setup

As of SevOne Data Insight 3.14, flag **FF_UI_REDIRECT** has been added to provide the capability to redirect the user interface using the *redirect* option in the URL.

URL Redirector API

The URL redirector is a URL-based API for translating simple resource information into complex Data Insight user interface data types for webhooks and other linking purposes.

Routing

The default redirector route is at **/redirect/v1/reports**. At present, **reports** are supported for redirection and URL translation. If you are not logged in, you will be prompted to log in and then, redirection will continue.

With the new URL redirector API (i.e., flag **FF_UI_REDIRECT**), you can easily create, view, and understand webhook drillback URLs. By default, this flag is set to **false**. In order to use the *FF_UI_REDIRECT* flag, you must set it to **true**.

1. **SSH** into your SevOne Data Insight machine and log in as **sevone** and at the **Password** prompt, enter **sevone**.

```
$ ssh sevone@<virtual machine IP address or hostname>
```

2. Using a text editor of your choice, edit /opt/SevOne/chartconfs/di_custom.yaml file to set flag, *FF_UI_REDIRECT*, and then, save it.

> ⚠ If /opt/SevOne/chartconfs/di_custom.yaml file does not exist, please create one and add the following to it.

```
ui:
  env:
    FF_UI_REDIRECT: true
```

3. Apply the change made to /opt/SevOne/chartconfs/di_custom.yaml file.

```
$ sevone-cli playbook up --tags apps
```

Before configuring a webhook containing drillback URL, a SevOne Data Insight report must exist; this will be the destination for the drillback and ideally, this report will be shared with all the users. This will ensure that all users who will click the link in the webhook, will be able to access the desired report.

The webhook URL sends the *resources* to the report. These resources are intended to provide context from the generating alert to the report to place a user, in context, within the report. It is highly recommended that the chosen report has variables configured for the report to accept these resources. From SevOne Data Insight, a report can be prepared for the resources that will be passed from the webhook URL. Using a web browser of your choice, enter your SevOne Data Insight IP address or hostname of the machine to which you want to add the report variables. Please refer to *SevOne Data Insight User Guide* > section Create > Report Options > Variables for details.

Example

> ⚠ The dynamic URL creation process references the selected resource types that are used in SevOne Data Insight OOTB (out-of-the-box) reports.

4.10.2 SevOne NMS Setup

Any SevOne NMS sending out webhooks containing drillback URLs must already be configured as a datasource on an instance of SevOne Data Insight. Administrators creating Webhook Definitions must already know the IP address / hostname or the URL of SevOne Data Insight instance and the names of any SevOne Data Insight reports which will be referenced in the drillback URL.

This feature is designed such that SevOne NMS clusters must have the same name as their SevOne Data Insight datasource name in multi-datasource use-cases.

To create a webhook definition, please refer to *SevOne NMS User Guide* > section **Webhook Definition Manager** > subsection **Add Webhook Definition** for details. In addition to this, you will also find details on how to **configure Slack channel**.

> Sections **Edit Webhook Definition** and **Delete Webhook Definition** provide details on how to edit or delete an existing webhook.

SevOne Data Insight drillback URLs are configured inside Webhook Definitions as part of the *message* body. Administrators can build their own definitions or reference the starter template for Slack message.

4.10.3 SevOne Data Insight Drillback URL Structure

To create a SevOne Data Insight drillback URL inside Webhook Definition, the following is required.

1. address of SevOne Data Insight instance which the URL will reference
2. name of SevOne Data Insight report the user will be navigated to
3. resources that will be referenced from the alert dynamically as variable **$DIDataResources**. This variable contains alert-specific information about the threshold violating the following.
    a. Datasource / Cluster Name
    b. Plugin Name
    c. Object Types
    d. Device Name
    e. Object Name(s)
    f. Indicator Type(s)
    g. Relative Timespan

Example: Create SevOne Data Insight URL in SevOne NMS Alert

Generic URL

http://<SevOne Data Insight IP address or hostname>/redirect/v1/reports?<enter parameter>&$DIDataResources

Example# 1

Let's assume the following.

- **SevOne Data Insight address or hostname**: localhost
- **SevOne Data Insight report name**: Alert Details Report
- **Resources**: this is automated by passing variable *$DIDataResources* in the URL

Your URL will be,

http://localhost/redirect/v1/reports?reportName=Alert%Details%Report&$DIDataResources

where, the parameter is **reportName**.

Example# 2

Let's assume the following.

- **SevOne Data Insight address or hostname**: 10.128.10.24
- **SevOne Data Insight report name**: Alert Details Report
- **Resources**: this is automated by passing variable *$DIDataResources* in the URL

Your URL will be,

http://10.128.10.24/redirect/v1/reports?reportName=Alert%Details%Report&$DIDataResources

where, the parameter is **reportName**.

⚠ For the list of available parameters, please refer to section Parameters.

Once you have created a valid SevOne Data Insight drillback URL, it will send the webhook's parameters to the Data Insight instance running at *localhost*. When this instance of Data Insight receives the URL, it will open *Alert Details Report* and automatically fill out the variables with the values provided by the webhook.

4.10.3.1 Single Datasource

This feature is designed to work with single NMS cluster as a datasource, with no additional configuration necessary.

4.10.3.2 Multi Datasource

To navigate users to the correct report resources in environments where there are multiple NMS clusters set up as Data Insight datasources, the Cluster Name of each NMS cluster sending webhook alerts must match the Data Insight datasource. *$DIDataResources* automatically appends the value of the Cluster Name as parameter, *datasourceName*, to the URL. In the event of a mismatch, the administrator can change to the desired datasource inside SevOne Data Insight. This design approach is chosen to require consistent naming and navigation to the correct resources.

Example: Functional URL

https://localhost/redirect/v1/reports?
reportName=Alert%20Details%20Report*&datasourceName=MyClusterName*
&indicators=SNMP%26%26Interface%26%26SNMPDevice%201%26%26Fa0%2F11%26%26ifInErrors%7C%7CSNMP%26%26Interface%26%26SNMPDevice%201%26%26Fa0%2F11%26%26ifOutErrors%7C%7CSNMP%26%26Interface%26%26SNMPDevice%201%26%26Fa0%2F11%26%26ifInDiscards%7C%7CSNMP%26%26Interface%26%26SNMPDevice%201%26%26Fa0%2F11%26%26ifOutDiscards&startTime=1667409663000

where,

Base URL is,

- **SevOne Data Insight address**: https://localhost/redirect/
- **API version**: v1
- **API function**: reports
- **Report Name**: Alert Details Report

Dynamic Alert Variables are,

- **Datasource / Cluster Name**: MyClusterName
- **DIDataResources**:
  - **Plugin**: SNMP
  - **Object Type**: Interface
  - **Device Name**: SNMPDevice 1
  - **Object Name**: Fa0/11

    - **Indicator Type**: ifInErrors
      OR
    - **Plugin**: SNMP
    - **Object Type**: Interface
    - **Device Name**: SNMPDevice 1
    - **Object Name**: Fa0/11
    - **Indicator Type**: ifOutErrors
      OR
    - **Plugin**: SNMP
    - **Object Type**: Interface
    - **Device Name**: SNMPDevice 1
    - **Object Name**: Fa0/11
    - **Indicator Type**: ifInDiscards
      OR
    - **Plugin**: SNMP
    - **Object Type**: Interface
    - **Device Name**: SNMPDevice 1
    - **Object Name**: Fa0/11
    - **Indicator Type**: ifOutDiscards
- **Relative Timespan**: 1667409663000 (as **Epoch timestamp**)

> **URL Legend**
>
> The URL ASCII character encoding has been omitted from URL legend above for readability.
> - %20 = space
> - %26 = &
> - %2f = /
> - %7C = |

4.10.3.3 Parameters

Parameters are provided in *query search* format.

```
http://localhost:8080/redirect/v1/reports?reportName=Indicator Summary&datasourceName=CX NMS
```

where,

- **reportName** = Indicator Summary
- **datasourceName** = CX NMS

URLs must be URL-encoded before being passed to the redirect route.

4.10.3.3.1 Combined Parameters

Some fields require information to be in a combined format and have optional support for multiple parameters of a type. In the case of a combined input, fields must be separated by **&&**. For example, **(Device Name)&&(Object Name)** for an *object* input.

Use **||** as a separator for fields with optional multiple parameters. For example, **(Device Name 1)&&(Object Name 1)||(Device Name 2)&&(Object Name 2)**.

4.10.3.3.2 reportId

Is the *report ID* parameter. It takes the report ID and sets it as the redirect target. This will override any ID passed in **reportName** parameter.

4.10.3.3.3 reportName

Is the *report name* parameter. It translates a report name into the report ID and sets it as the target of the redirect when using the reports redirect route.

4.10.3.3.4 datasourceId

Is the *datasource ID* parameter. It takes a given datasource ID and sets it as the target datasource. It overrides any provided **datasourceName** parameter.

4.10.3.3.5 datasourceName

Is the *datasource name* parameter. It translates a datasource name into an ID and sets it as the target datasource.

4.10.3.3.6 startTime / endTime

Are the *specific time* parameters where **startTime** and **endTime** are UNIX timestamps. It takes a UNIX timestamp *startTime* and optional *endTime* and sets it as the timespan for the report. If no endTime is provided, it defaults to the current UNIX timestamp. This will always override both *timespan* and *customTimespan* parameters.

4.10.3.3.7 timespan

Is the *SevOne timespan* parameter. It takes a SevOne timespan for example, **PAST_24HOURS**, and sets it as the timespan for the report.

4.10.3.3.8 customTime

Is the *custom time* string parameter. It takes a custom time string for example, **PAST 24 Hours**, and sets it as the timespan for the report.

4.10.3.3.9 timezone

Is the *timezone* string parameter. It takes a timezone string and applies it to the **timespan** parameter. For example, **America/Anchorage**.

4.10.3.3.10 deviceGroups

Is the *[Device Group Path]* parameter separated by **/**. It takes a set of device group paths and translates them into device groups useable by report variables.

4.10.3.3.11 devices

Is the *[Device Name]* parameter. It takes a set of device names and translates them into devices useable by report variables.

4.10.3.3.12 objects

Is the *[Plugin Name + Device Name + Object Name]* parameter. It takes a set of plugin, device and object names and translates them into objects useable by report variables. At present, the object report variable only uses the first object.

4.10.3.3.13 indicators

Is the *[Plugin Name + Object Type Path (separated by /) + Devcie Name + Object Name + Indicator Type Name]* parameter. It takes a set of plugin, object type, device, object and indicator type names and translates them into indicators useable by report variables.

5 Deployment

5.1 Backup and Restore

5.1.1 Backup

It is always a good idea to regularly back up your **report data** and **decryption keys**. Please execute the following steps to **back up**.

```
$ mkdir -p /home/sevone/backups

$ sevone-cli exec mysql -- mysqldump -uroot -pdatainsight \
--lock-tables=false --databases datainsight reportscheduler | \
grep -v "Using a password" > \
/home/sevone/backups/mysqldump-$(date +%d-%m-%Y-%s).sql

$ kubectl get secret securitykeys -ojson | \
jq -r '.data."keys.json"' | \
base64 -d > /home/sevone/backups/keys.json
```

5.1.2 Teardown

In some cases, it may be necessary to teardown the Kubernetes cluster.

```
$ sevone-cli cluster down
```

5.1.3 Restore

Report data and decryption keys can only be restored on a running Kubernetes cluster.

```
$ sevone-cli cluster up
```

```
$ MYSQL_STS=$(kubectl get sts -l app.kubernetes.io/name=mysql -oname)
$ SECRET=$(kubectl get secrets | grep securitykeys | awk '{print $1}')

$ kubectl rollout status -w $MYSQL_STS

$ kubectl delete secret $SECRET

$ kubectl create secret generic $SECRET \
--from-file=keys.json=/home/sevone/backups/keys.json

$ sevone-cli exec mysql -- mysql -uroot -pdatainsight \
< $(ls -Art ~/backups/mysqldump-*.sql | tail -n 1)
```

```
$ kubectl delete pods -l app.kubernetes.io/component=graphql
```

5.2 Change Hostnames

5.2.1 Teardown Kubernetes

> Whenever you teardown the cluster be sure to back up your data first.

To change a node's hostname, you must teardown your Kubernetes cluster.

```
$ sevone-cli cluster down
```

5.2.2 Update ansible Inventory

1. Run the following command on *every* node to change their hostname.

   Example

   ```
   $ sudo hostnamectl set-hostname "sdi-node-01"
   ```

2. On the *control plane* node, update **/etc/ansible/hosts** with your new hostname.

   Example

   ```
   [server]
   sdi-node-01 ansible_connection=local
   ```

3. If you have *agent* nodes, update their hostnames as well.

   Example

   ```
   [server]
   sdi-node01 ansible_connection=local

   [agent]
   sdi-node02 ansible_host=10.123.45.68
   sdi-node03 ansible_host=10.123.45.69
   ```

5.2.3 Provision Kubernetes

If you have torn down a running cluster, after executing the command below, you must restore your data.

```
$ sevone-cli cluster up
```

5.3 Handle IP Conflicts

The following are the default IP ranges used by SevOne Data Insight.

Flag	Description	IP Address	IP Range
--cluster-cidr	Pod IP addresses	192.168.80.0/20	192.168.80.0.0 - 192.168.95.255
--service-cidr	Service IP addresses	192.168.96.0/20	192.168.96.0 - 192.168.111.255
--cluster-dns	Cluster DNS (must be in Service's range)	192.168.96.10	n/a

5.3.1 Teardown Kubernetes

Whenever you teardown the cluster be sure to back up your data first.

1. In order to change the default IP ranges, you must teardown your Kubernetes cluster.

```
$ sevone-cli cluster down
```

2. Ensure that the old IP address ranges are not left behind in any of your node's routing tables.

```
$ ansible all --become -a "ip route del 192.168.96.0/24"
```

5.3.2 Adjust IP Ranges

Create a file **ip_ranges.yaml** in **/etc/ansible/group_vars/all** directory with your *new* IP ranges.

Example

```
$ echo 'k3s_cluster_cidr: "10.42.0.0/16"' >> \
/etc/ansible/group_vars/all/ip_ranges.yaml

$ echo 'k3s_service_cidr: "10.43.0.0/16"' >> \
/etc/ansible/group_vars/all/ip_ranges.yaml

$ echo 'k3s_cluster_dns: "10.43.0.10"' >> \
/etc/ansible/group_vars/all/ip_ranges.yaml
```

You may then redeploy or proceed with your deployment as normal.

```
$ sevone-cli playbook up
```

5.4 Handle NMS Failover

SevOne Data Insight supports NMS Failover. This process is completely automated and is setup during installation if the PAS and HSA pair is set.

⚠ Failover is supported only when the peers are configured with IPv4 addresses. At present, IPv6 is not supported.

5.5 Rotate Kubernetes Certificates

During SevOne Data Insight upgrade, the k3s service automatically rotates certificates that are due to expire within 90 days. In the event that they expire before k3s is able to rotate them, you will need to rotate manually.

```
$ kubectl get pods
Unable to connect to the server: x509: certificate has expired or is not yet valid
```

5.5.1 Backup TLS Directory

As a precautionary measure, backup the TLS directory.

```
$ sudo tar -czvf /var/lib/rancher/k3s/server/tls.tgz /var/lib/rancher/k3s/server/tls
```

5.5.2 Generate New Certificates

1. Remove the cached certificate from a Kubernetes secret.

```
$ sudo rm /var/lib/rancher/k3s/server/tls/dynamic-cert.json
```

2. Restart k3s service to rotate the certificates.

```
$ sudo systemctl restart k3s
```

> You can now run Kubernetes commands. This will allow you to backup your all-important security keys in case you have not done so already.

5.5.3 Refresh Kubernetes Config

After rotating the Kubernetes certificates, the Kubernetes configuration file must be refreshed to apply the new certificates.

Refresh Kubernetes config file

for 'root' user

```
$ sudo cp /etc/rancher/k3s/k3s.yaml /root/.kube/config
```

for 'sevone' user

```
$ sudo cp /etc/rancher/k3s/k3s.yaml /home/sevone/.kube/config
$ sudo chown -R sevone:sevone /home/sevone/.kube
```

5.5.4 Verify Certificates

To verify the certificates, execute the following commands.

```
$ sudo -i

$ for i in `ls /var/lib/rancher/k3s/server/tls/*.crt`; \
do echo $i; openssl x509 -enddate -noout -in $i; \
echo "---"; done
```

6 References

AWS CloudWatch	• https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/create-cloudwatch-agent-configuration-file.html
Helm Package Manager	• https://helm.sh/
Kubernetes	• https://kubernetes.io/ • https://kubernetes.io/docs/tasks/access-application-cluster/list-all-running-container-images/
Kubernetes Resource Requests & Limits	• https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/#requests-and-limits