Turbonomic 7.22.5 Installation Guide (v2)

Turbonomic, Inc

500 Boylston St, 7th floor
Boston, MA 02116 USA
Phone: (844) 438-8872
www.turbonomic.com

COPYRIGHT

END-USER LICENSE AGREEMENT

https://cdn.turbonomic.com/wp-content/uploads/Turbonomic_Click_Through_Customer-License.pdf

Contents

Introduction

Thank you for choosing Turbonomic, the Intelligent Workload Automation Management solution for Virtualized Environments. This guide gives you information you need to install Turbonomic in your virtual environment, install your license, and get started managing your resources.

If you have any questions, please contact Turbonomic support. Visit our support site at `https://support.turbonomic.com`.

Sincerely:

The Turbonomic Team

Minimum Requirements

License Requirements

To run Turbonomic on your environment, you must install the appropriate license. Licenses enable different sets of Turbonomic features, and they support a specified number of workloads in your environment.

This version of Turbonomic requires workload-based licenses. The platform does not support socket-based licenses.

User Interface Requirements

To display the Turbonomic user interface, you must log into the platform with a browser that is capable of displaying HTML5 pages.

Network Addressing Requirements

Turbonomic requires static IP addressing. Static IP setup is covered as a step when installing the Turbonomic OVA.

Compute and Storage Requirements

The requirements for running a Turbonomic instance depend on the size of the environment you are managing. Turbonomic keeps a real-time representation of your environment in memory. The greater the number of entities to manage, and the greater the relationships between them, the more resources you need for the VM that runs Turbonomic. And as the VM requirements increase, so do the requirements for the physical machine that hosts the VM.

The requirements listed here are recommendations that you should keep in mind as you plan your Turbonomic deployment. After deploying, if you find that you need to change memory capacity, CPU capacity, or both for the VM, you can shut it down, make changes, and then power it up again to use the new capacity.

NOTE:
The machine that hosts the Turbonomic platform must support the SSE4.2 instruction set. Support for this instruction set was introduced at different times for different chip manufacturers:

- Intel: November 2008
- AMD: October 2011

The machine you use to host Turbonomic should be newer than these dates. On a Linux system, you can run the following command to check for this support:

`cat /proc/cpuinfo | grep sse4`.

For more information, see the glossary entry at `http://www.cpu-world.com/Glossary/S/SSE4.html`.

In most cases you can run Turbonomic on a host that meets the following minimum requirements:

Supported Hypervisors		Storage Requirements	Memory	CPUs
VMware	vCenter versions 5.5, 6.0, 6.5, 6.7, and 7.0	1.25 TB or greater. **NOTE:** Can be thin provisioned depending on the storage requirements.	• Default: 128 GB • For 10,000 VMs or less, 64 GB	8 vCPUs

Turbonomic provides an OVA file which is preconfigured with two hard drives. A minimum of 1.25 TB is necessary to ensure that the drives have the proper amount of space for storage.

Installing on VMware Systems

This download of the Turbonomic instance is in the `.OVA 1.0` format.

> **NOTE:**
> For minimum requirements, we recommend 128 GB of memory for the VM that hosts Turbonomic. However, if you plan to manage a smaller environment (10,000 VMs or less), you can install on a VM that provides 64 GB of memory. (See Minimum Requirements *(on page 5)*).
>
> If you plan to install a VM with 64 GB of memory, then you must:
>
> - Modify the default for VM memory (see Deploy the Turbonomic VM. *(on page 7)*)
> - Set the `charts_v1alpha1_xl_cr.yaml` file that you will use to deploy the Kubernetes nodes for the Turbonomic platform (see Optionally, set the installation to the smaller memory requirement of 64 GB *(on page 10)*).

To install Turbonomic:

1. Download the Turbonomic installation package.

   The email you received from Turbonomic includes links to the Turbonomic download pages. You can get the installation package from there.

   The installation package includes the `turbonomic_t8c-<version>-<XXXXXXXXXXXXXX>.ova` file

   where <version> is the Turbonomic version number and <XXXXXXXXXXXXXX> is the timestamp.

   For example: `turbonomic-t8c-7.17.3-20190916164429000.ova`

   The OVA file deploys as a VM with the Turbonomic components already installed.
2. Import the OVA file into your datacenter.

   Use the vCenter Server client to import the OVA into your environment.
3. Deploy the Turbonomic VM.

   Create the VM using the OVA file. Ensure that the physical machine hosting the VM meets the minimum requirements (see Minimum Requirements *(on page 5)*).

   For minimum requirements, we recommend 128 GB of memory for the VM that hosts Turbonomic. However, if you plan to manage a smaller environment (10,000 VMs or less), you can install on a VM that provides 64 GB of memory. (See Minimum Requirements *(on page 5)*).

If you want to deploy a VM with 64 GB of memory, manually modify the default value for Memory:

a. Right-click the VM and choose **Edit Settings**.
b. Type **64** for Memory.
c. Click **OK** to save the settings
d. Power on the VM.

4. Open the remote console.

   For the Turbonomic VM that you just deployed:

   a. Choose the **Summary** tab.
   b. Click **Launch Remote Console**.

5. Set up the Turbonomic System Administrator account.

   a. In the remote console, log in with the following default credentials:

      - Username: `turbo`

        Do not use the account name, `root`.
      - Password: `vmturbo`

        Then, you will be prompted to enter a new password.

   b. Enter your new password.

      The new password must comply with the strong password policy (a mixture of upper- and lower-case letters, numbers, and a symbol). Only you will know this new password.

      > **NOTE:**
      > *Be sure to save the turbo account credentials in a safe place.* For security reasons, this is the only account that can access and configure the Turbonomic VM.

   c. Enter your new password again to verify it.

6. Update the root password.

   The platform uses the `root` account for certain processes, such as rolling up log messages in `/var/log/messages`. To ensure the account credentials are current, you must change the password:

   a. Open a SuperUser session.

      - In the remote console, enter `su -`
      - At the password prompt, enter `vmturbo`

   b. Reset a new password.

      The new password must comply with the strong password policy (a mixture of upper- and lower-case letters, numbers, and a symbol). Only you will know this new password.

      > **NOTE:**
      > *Be sure to save the root account credentials in a safe place.*

      - Enter the command, `passwd`
      - At the new password prompt, enter the new password
      - At the Retype prompt, enter the password again

        You should see the message `password updated successfully`.

   c. Exit the SuperUser session.

Enter `exit`.

7. Set up the static IP address.
   a. In the remote console, start the NetworkManager user interface.

      Type: `sudo nmtui`
   b. Edit the `eth0 ethernet` connection.

      Use the arrow keys to navigate in NetworkManager.

      In NetworkManager, choose the `Edit` option and click <OK> to open the Edit Connection dialog box.

      Then, choose `Ethernet` from the list of options and fill out the following information (values given are examples only).

      - Profile Name: `System eth0`
      - Device: `eth0`

      Next, expand `IPv4 CONFIGURATION` and fill out the configuration subsettings based on your environment:

      - IPv4 Configuration: <Manual>
      - Addresses: Specify the static IP address you want, with the subnet mask on the same line. For example:

        `10.0.254.10/24`
      - Gateway, DNS Servers, and Search domains: Give values that are valid for your network environment.

      When you are satisfied with your settings, click <OK> and <Back> to return to the configuration list. Verify that the connection you just created is present.
   c. Make a note of the IP addres that you just specified.
   d. Exit NetworkManager.

      Click <Quit> to return to the command line.
8. Restart the network service in the Turbonomic VM.

   Type: `sudo systemctl restart network`
9. Optional: Set up a proxy for internet access.

   To use a proxy for internet access, you should set it up before you install the Kubernetes nodes for the Turbonomic platform.

   > **IMPORTANT:**
   > Do not configure your proxy in the `/etc/environment` file. This configuration is for the host machine, and will interfere with Kubernetes networking. Instead, you must configure a proxy for the Kubernetes environment. The following instructions are to set up the installation process so it will configure your proxy upon install.
   >
   > To verify that a proxy has not been configured for the host machine, open `/etc/environment`. The file should have no content. If a proxy has been configured in this file, delete the configuration statements and save your changes. Then log out of the host machine and log in again.

   To set up a proxy for this installation:

   a. Note the IP address of your host machine.

      When you set the static IP for this machine, you should have made a note of the IP address. If you do not know the machine's IP address, execute the following command:

      `ifconfig eth0`
   b. Edit the Kubernetes setup so that `packagemanager` and `docker daemon` will use your proxy correctly.

Open the following file for edit:

`/opt/kubespray/inventory/sample/group_vars/all/all.yml`

c. Specify the proxy to use.

Search for `http_proxy`. You should see the following two lines, commented out:

```
# http_proxy:
# https_proxy:
```

Uncomment the line for the protocol your proxy uses, and add the the proxy address and port. The most common protocol is HTTP, so you would uncomment `http_proxy` and specify your proxy server. For example:

```
http_proxy: "10.10.12.34:3123"
```

d. Exclude this host VM from going through the proxy server.

Proxy configuration is to set up the Turbonomic components to go through the proxy. However, you should exclude the host machine.

In the same `/opt/kubespray/inventory/sample/group_vars/all/all.yml` file, search for `no_proxy`. Uncomment this line and specify the host VM. Use the IP address that you specified when you set a static IP. The line should read:

```
no_proxy: "<YourStaticIP>, node1, 127.0.0.1, 127.0.0.0"
```

Where `<YourStaticIP>` is the IP address you specified for the Turbonomic VM.

e. Save your changes.

When you are done, save and close the file. This sets up the installation process to configure the proxy for the Turbonomic components.

f. Update your `yum` configuration to use the proxy.

If you set up a proxy for your Turbonomic platform, then you should also set the same proxy for `yum`, so you can properly download and install component upgrades.

Open the file, `/etc/yum.conf` and search for `# proxy=`. Uncomment the line and add your proxy server IP and port. For example, the line should be similar to:

`proxy=10.10.12.34:3123`.

When you are done, save and close the file.

To verify that your proxy has been set, after you run the installation you can view the configuration in `/etc/systemd/system/docker.service.d/http-proxy.conf`. Find the `[Service]` section. It should contain your `proxy` and `no_proxy` settings. For example, it should be similar to:

```
Environment="HTTP_PROXY=http://10.10.12.34:3123" "NO_PROXY=<YourStaicIP>, node1,
127.0.0.1, 127.0.0.0"
```

If you want to change the proxy configuration after you have installed Turbonomic, please contact Technical Support.

10. Optionally, set the installation to the smaller memory requirement of 64 GB.

For minimum requirements, we recommend 128 GB of memory for the VM that hosts Turbonomic. However, if you plan to manage a smaller environment (10,000 VMs or less), you can install on a VM that provides 64 GB of memory. (See Minimum Requirements *(on page 5)*).

To set the smaller installation, you will set a symbolic link to the `charts_v1alpha1_xl_cr.yaml` file that you want to use:

- In a shell, `cd` to the `deploy/crds` directory:

  ```
  cd /opt/turbonomic/kubernetes/operator/deploy/crds
  ```

- Remove the default symbolic link.

  By default, the symbolic link points the the 128 GB installation script. Use the command:

  ```
  rm charts_v1alpha1_xl_cr.yaml
  ```

- Create a symbolic link to the 64 GB installation script:

  ```
  ln -s charts_v1alpha1_xl_cr-64gb.yaml charts_v1alpha1_xl_cr.yaml
  ```

11. Optionally, configure Single Sign-On Authentication (SSO) for this installation.

    To configure SSO, you must edit the `charts_v1alpha1_xl_cr.yaml` file. You can edit it now, before you complete the installation, or you can edit it later and restart the affected components. For more information, see Single Sign-On Authentication *(on page 34)*.

12. Deploy Turbonomic Kubernetes nodes.

    When you deploy Turbonomic on Kubernetes, you deploy one Kubernetes node as a VM that will host pods to run the Turbonomic components. The script to deploy and initialize the Kubernetes node also deploys the Kubernetes pods that make up the Turbonomic application.

    Start a secure session (SSH) on your Turbonomic VM as the turbo user and perform the following steps:

    a. Initialize the Kubernetes node and deploy the pods.

       Execute the script: `/opt/local/bin/t8c.sh`

       Do not specify `sudo` when you execute this script.

       The script should take up to 20 minutes to complete.

    b. Verify that the the deployment succeeded.

       At the end of the script output, in the summary section, verify that no errors are reported. If any errors are reported, contact Turbonomic Support.

    c. Verify that the Turbonomic application installed correctly.

       To verify the installation of the application, execute the command:

       ```
       kubectl get pods -n turbonomic
       ```

       After all of the pods start up, the READY column should read `1/1` and the STATUS column should read `Running`.

       You should see output similar to the following:

       ```
       NAME                                  READY   STATUS    RESTARTS
       action-orchestrator-b6454c9c8-mfl85   1/1     Running   0
       api-7887c66f4b-shndq                  1/1     Running   0
       arangodb-7f646fc5fc-zhcwf             1/1     Running   0
       auth-5b86976bc8-vxwz4                 1/1     Running   0
       clustermgr-85548678d9-r5wb8           1/1     Running   0
       consul-7f684d8cb8-6r677               1/1     Running   0
       cost-5f46dd66c4-6d6cb                 1/1     Running   0
       ```

```
group-5bfdfbc6f8-96bsp                          1/1     Running   0
history-5fc7fbc855-6zslq                        1/1     Running   0
kafka-74cc77db94-dfrbl                          1/1     Running   0
market-5f54699447-z4wkm                         1/1     Running   0
mediation-actionscript-57b4fc6df-4lzfv          1/1     Running   0
mediation-appdynamics-6d65f8766f-kb44l          1/1     Running   0
mediation-hpe3par-d7c475c4c-v8ftc               1/1     Running   0
mediation-hyperv-6bd8c94df5-4dbzx               1/1     Running   0
mediation-netapp-7f8fc955d9-4kkdl               1/1     Running   0
mediation-oneview-7dbd7b54cf-7rfqp              1/1     Running   0
mediation-pure-58c4bd8cd9-8n256                 1/1     Running   0
mediation-ucs-6f4bb9889-9rnqk                   1/1     Running   0
mediation-vcenter-5bc4f5fbd4-nzm4j              1/1     Running   0
mediation-vcenterbrowsing-5c5987f66c-bfjq4      1/1     Running   0
mediation-vmax-6c59969b89-28t9j                 1/1     Running   0
mediation-vmm-9c4878cf9-rfxnl                   1/1     Running   0
nginx-5b775f498-sm2mm                           1/1     Running   0
plan-orchestrator-6dffc4c9b6-p5t5n              1/1     Running   0
repository-6d555bb4bf-fxldh                     1/1     Running   0
rsyslog-fd694878c-5tb2c                         1/1     Running   0
t8c-operator-558bcc758d-5h8mp                   1/1     Running   0
topology-processor-b646b786b-9skp7              1/1     Running   0
zookeeper-5f65b5bf69-nnmbt                      1/1     Running   0
```

d. Synchronize the system clock.

   To ensure correct display of data, and to support Single Sign-On (SSO) authentication, you need to synchronize the system clock.

   For information, see Synchronizing Time *(on page 15)* and Single Sign-On Authentication *(on page 34)*.

e. Verify that the Load Balancer has installed correctly.

   To verify the presence of the Load Balancer, execute the command:

   ```
   kubectl get services -n turbonomic|grep LoadBalancer
   ```

   You should see output similar to the following:

   ```
   nginx  LoadBalancer   10.10.10.10   10.10.10.11  443:32669/TCP,80:32716/TCP  17h
   ```

f. Enable mediation.

   The t8c.sh script automatically enables mediation. No user action is required.

   For Turbonomic to manage your IT environment, it must attach to targets in your environment so it can perform discovery and execute actions. The combination of the processes of discovery and action execution is *mediation*. This release of Turbonomic on Kubernetes supports mediation through the following targets. If you need to use additional targets that are not in this list, contact Turbonomic Support.

   - Applications and Databases
     - Microsoft SQL Server 2008 R2, 2012, 2014, and 2016
     - AppDynamics 4.1+
     - DynaTrace 1.1+
     - AppInsights 4.1+

- Cloud Native Targets
  - OpenShift 3.3+
  - Kubernetes
- Fabric and Network
  - Cisco UCS Manager 3.1+
  - HPE OneView 3.00.04+
- Guest OS Processes
  - SNMP
  - WMI: Windows versions 2019, 2016, 2012 / 2012 R2, 2008 R2, 10, 8 / 8.1, and 7
- Hyperconverged
  - Nutanix Community Edition
- Hypervisors
  - VMware vCenter 5.1, 5.5, 6.0, 6.5, 6.7, and 7.0
  - Microsoft Hyper-V 2008 R2, Hyper-V 2012, and Hyper-V 2012 R2
- Orchestrator Targets
  - Action Script
- Private Cloud Managers
  - Microsoft System Center 2012 Virtual Machine Manager and System Center 2012 R2 Virtual Machine Manager
- Public Cloud Managers
  - Amazon AWS
  - Amazon AWS Billing
  - Microsoft Azure
  - Microsoft Enterprise Agreement
- Storage Managers
  - Pure Storage F-series and M-series arrays
  - NetApp Cmode/7mode using ONTAP 8.0+ (excluding AFF and SolidFire)
  - EMC VMAX using SMI-S 8.1+
  - EMC ScaleIO 2.x and 3.x
  - HPE 3PAR InForm OS 3.2.2+, 3PAR SMI-S, 3PAR WSAPI
- Virtual Desktop Infrastructure
  - VMware Horizon

For information about these targets, see the *Turbonomic Target Configuration Guide*.

13. Log in to the Turbonomic user interface and set the administrator user account password.

> **IMPORTANT:**
> You should wait until all the platform components have started up, are running, and are fully ready before your first login. If you try to add a license or add a target to the platform before the components are all ready, the platform can fail to initialize correctly. For more information, see Verify that the Turbonomic application installed correctly. *(on page 11)*

After the components start up, in your Web browser, type the static IP address of your Turbonomic VM. Your browser redirects to `https://[MyIPAddress]/app/index.html#/authentication/login`. This is the login page for Turbonomic users.

Turbonomic includes a default user account named `administrator`. You cannot delete this account, and you must set your own password for it.

In the login page, enter the information as required, and make a note of it.

- Use the default credential for **USERNAME**: `administrator`.
- Type a password for **PASSWORD**.

  The new password must comply with the strong password policy (a mixture of upper- and lower-case letters, numbers, and a symbol). Only you will know this new password.
- Type the password again to verify it for **REPEAT PASSWORD**.
- Click **CONFIGURE**.

This is the account you will use to access the Turbonomic user interface with administrator permissions. *Be sure to save the user interface administrator account credentials in a safe place.*

> **NOTE:**
> The initial login is always for the `administrator` account. This is an administration *user* account. Do not confuse this with the Turbonomic System Administrator account that you previously set up to log into shell sessions on the VM itself.

14. After you have logged in as `administrator`, you can create other user accounts, and you can give them various roles. For more information about user accounts and roles, see the *Turbonomic User Guide*.

General Configuration Tasks

After you install the Turbonomic instance, perform the following configuration tasks:

- (Required) Synchronize the system clock and configure your time servers.
- (Optional) Enforce secure access by installing a trusted certificate.

(Required) Synchronizing Time

It is important that you synchronize the clock on the Turbonomic instance with the other devices on the same network. By default, the Turbonomic server is configured to synchronize with any one of the following time servers:

- `0.centos.pool.ntp.org`
- `1.centos.pool.ntp.org`
- `2.centos.pool.ntp.org`
- `3.centos.pool.ntp.org`

To synchronize with these servers, your installation of Turbonomic must have access to the internet. If your environment restricts internet access, then you have to configure synchronization with a time server on your network.

In all cases, you should verify that the Turbonomic clock is properly synchronized. To check the system clock:

1. Open a SSH terminal session to your Turbonomic instance.

   Log in with the System Administrator that you set up when you installed Turbonomic:

   - Username: `turbo`
   - Username: `[your_private_password]`

2. Verify your time settings.

   Execute the `date` command. You should see results similar to:

   ```
   Thu Feb 2 14:25:45 UTC 2019
   ```

   To verify the time, compare the `date` output with the output from a known UTC time server.

   Alternately you can execute the command, `timedatectl`. The output should be similar to:

   ```
         Local time: Fri 2019-12-06 21:09:26 UTC
     Universal time: Fri 2019-12-06 21:09:26 UTC
           RTC time: Fri 2019-12-06 21:09:27
          Time zone: UTC (UTC, +0000)
        NTP enabled: yes
   NTP synchronized: yes
    RTC in local TZ: no
         DST active: n/a
   ```

   This tells you whether you have NTP enabled, and whether it is currently synchronized, along with other time synchronization information.

If the output is correct *and* your environment has access to the internet, you can assume the system clock is synchronized.

If the output is incorrect, or if you need to configure synchronization with a time server on your network, you must configure `chrony` on the server instance.

To set up `chrony` on your Turbonomic instance:

1. Open an SSH terminal session to your Turbonomic instance.
2. Open the `chrony` configuration file.

   For example, execute the command: `sudo vi /etc/chrony.conf`
3. Specify the time servers that you want to use in your environment.

   The `chrony` file includes the following statements to configure time servers:

   ```
   server 0.centos.pool.ntp.org iburst
   server 1.centos.pool.ntp.org iburst
   server 2.centos.pool.ntp.org iburst
   server 3.centos.pool.ntp.org iburst
   ```

   Enter statements for the servers you want to use. Then delete or comment out the statements that you do not want to use.

   Specify a time server via the following command syntax:

   `server` *`My_Time_Server_Name`* `iburst`
4. Save the file.
5. Restart the chrony service.

   Execute the command: `sudo systemctl restart chronyd`
6. Verify that your time is correct.

   Execute the `date` command. You should see results similar to:

   ```
   Thu Feb 2 14:25:45 UTC 2019
   ```

To verify the time, compare the `date` output with the output from a known UTC time server.

If the output is correct you can assume the system clock is synchronized.

If the output is incorrect, contact your support representative.

(Optional) Enforcing Secure Access Via LDAP

If your company policy requires secure access, you can use a certificate with you LDAP service to set up secure access for your users. For example, you can configure Active Directory (AD) accounts to manage *External Authentication* for users or user groups. The user interface to enable AD includes a **Secure** option, which enforces certificate-based security. For more information, see "Managing User Accounts" in the *Turbonomic User Guide*.

If your LDAP service uses a Certificate Authority (CA), then the certificate signed by that CA should support this feature as it is. Simply turn on the **Secure** option when you are setting up your AD connection.

If your LDAP service uses a self-signed certificate, then you must install that certificate on the Turbonomic authorization pod. The steps you will perform include:

- Get the certificate from your LDAP server
- Import the certificate to the platform's TrustStore
- Add the certificate to the Turbonomic platform's authorization pod
- Enable the TrustStore in the Turbonomic platform's Operator chart

This section describes how to set up secure access from an LDAP server. It assumes you have authorization to get a certificate from the LDAP server, as well as admin authority on the Turbonomic platform.

To set up secure access:

1. Open an SSH terminal session to your Turbonomic instance.

   Log in with the System Administrator that you set up when you installed Turbonomic:

   - Username: `turbo`
   - Password: `[your_private_password]`
2. Download your LDAP Server certificate to the Turbonomic instance.

   Acquire a certificate from your LDAP administrator, and download it to the Turbonomic platform. For example, you can download it to the file `/tmp/ldapserver.pem`:
3. Import the `.pem` file to the Turbonomic TrustStore.

   This step modifies the `cacerts` file on the Turbonomic platform.

   > **NOTE:**
   > To import a certificate to the Turbonomic TrustStore, you must use the `keytool` utility. To install this utility, execute the command:
   >
   > ```
   > sudo yum install java-1.8.0-openjdk
   > ```
   >
   > This installs the utility in `/usr/bin/keytool`.

   If an alias for an LDAP certificate already exists, delete that certificate. For example, assuming the alias `ldapcert1`, execute the following command:

```
keytool -delete -alias ldapcert1 -keystore cacerts -storepass changeit
```

Then use the following command to import your new certificate to the TrustStore:

```
keytool -import -alias ldapcert1 -file /tmp/ldapserver.pem -keystore cacerts
-deststoretype jks -storepass changeit -noprompt
```

4. Add the TrustStore to the Turbonomic authorization pod.

   Execute the following command to copy the `cacerts` file into the authorization pod:

   ```
   kubectl cp cacerts $auth_pod:/home/turbonomic/data/cacerts
   ```

5. Update the platform's Operator Chart to use the TrustStore.

   a. Open the chart file for editing.

      Open the file, `/opt/turbonomic/kubernetes/operator/deploy/crds/charts_v1alpha1_xl_cr.yaml`.

   b. Add the TrustStore as an authorization spec for the component options.

      In the chart file, find the `spec:` section. Within that section, find the `auth:` subsection.

      This should be the second subsection in `spec:`, after `global:`. If there is no `auth:` subsection, you can add it to `spec:`.

   c. Add the TrustStore to the `auth:` subsection.

      You will add the TrustStore path to a `javaComponentOptions:` statement within the `auth:` subsection. Add the path as a `-D` option. Use the same path that you copied the `cacerts` file to in the Turbonomic authorization pod. In the above example, you copied it to `$auth_pod:/home/turbonomic/data/cacerts`.

      Following the above example, the `auth:` subsection should be similar to the following:

      ```
      # Pass in the JAVA_OPTS to the auth POD to set up additional options such as
      # a trustStore for AD Certificate(s) for LDAPS (Secure LDAP)
      auth:
       javaComponentOptions: "-Djavax.net.ssl.trustStore=/home/turbonomic/data/cacer
      ts"
      ```

   d. Save the `charts_v1alpha1_xl_cr.yaml` file.

6. Apply your Operator Chart changes to the Turbonomic platform.

   Execute the following command:

   ```
   kubectl apply -f
   /opt/turbonomic/kubernetes/operator/deploy/crds/charts_v1alpha1_xl_cr.yaml
   ```

   This restarts the authorization component so it can use the new setting. As the component restarts, the `rsysloc` output should include the following Message:

   ```
   -Djavax.net.ssl.trustStore=/home/turbonomic/data/cacerts
   ```

(Optional) Enforcing Secure Access Via Trusted Certificate

If your company policy requires SSL connections via trusted certificate, Turbonomic enables you to install a trusted certificate from a known certificate authority.

Requesting a Certificate

The first step is to acquire a certificate. The following steps describe how to generate a certificate request.

1. Open a shell terminal session.

   Open an SSH terminal session on your Turbonomic instance. Log in as `turbo`, and use the password that you created for the administration account in the installation steps above. For information, see the installation step, Set up the Turbonomic System Administrator account *(on page 8)*.
2. Change to the directory where you want to store the private key file.

   If your shell session is on your Turbonomic instance, you should use the `/opt/turbonomic` directory:

   ```
   cd /opt/turbonomic
   ```
3. Create and save the private key file.

   Execute the command to create a private key file.

   For this example, the private key file is named `myPrivate.key`

   ```
   openssl genrsa -out myPrivate.key 2048
   ```

   You will need this file later. If you are in a session on your Turbonomic instance, you might want to copy the file to your local machine.
4. Create a file to contain the information that will generate the signed certificate request (CSR).

   ```
   vi certsignreq.cfg
   ```
5. Add the request data to the `certsignreq.cfg` file.

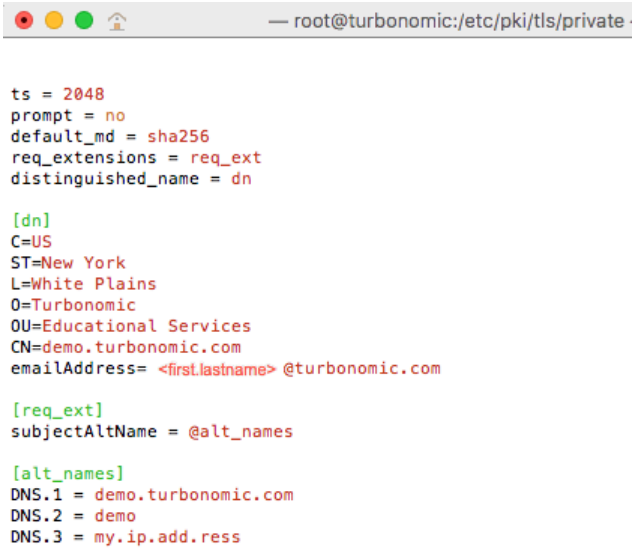
   In the file, insert the following code. For any fields marked by angle brackets (for example `<city>`), provide the indicated value. For example, your country, city, company, etc.

   ```
   [req]
   ts = 2048
   prompt = no
   default_md = sha256
   req_extensions = req_ext
   distinguished_name = dn

   [dn]
   C=<country, 2 letter code>
   L=<city>
   O=<company>
   OU=<organizational unit name>
   CN=<FQDN>
   emailAddress=<email address>

   [req_ext]
   ```

```
subjectAltName = @alt_names

[alt_names]
DNS.1 = <FQDN>
DNS.2 = <server's short name>
DNS.3 = <server's IP address>
```

> **NOTE:**
> For the `CN` field, specify the fully-qualified domain name of the Turbonomic instance.
>
> Alternate names are other ways to access the Turbonomic instance. In the `[alt_names]` section, the value for the `DNS.1` field is required. For `DNS.1`, specify the fully-qualified domain name of the Turbonomic instance. Values for the `DNS.2` and `DNS.3` are optional. You can add more DNS.<n> fields if needed.

For example:



6. Write and quit the file.

   Press **esc**, type `:wq!`, and press **Enter**.

7. Generate the certificate request file.

   In this example, we name the file `myRequest.csr`.

   Execute the command:

   ```
   openssl req -new -sha256 -nodes -out myRequest.csr -key myPrivate.key -config certsignreq.cfg
   ```

8. Send the generated request file to your certificate authority.

   If you generated the file on your Turbonomic instance, you should transfer the file to your local machine. The path to the certificate request file on your remote machine is `/opt/turbonomic/myRequest.csr`.

   Your certificate authority will use this file to create the certificate for you.

   If your certificate authority gives you an encoding choice between DER and Base 64, choose **Base 64**.

9. When you receive the certificate, save it to disk.

   If you did not receive the certificate encoded in Base 64, you must convert it from DER to Base 64. Execute the following command, assuming the certificate is named `MyCertificate.crt`:

   ```
   openssl x509 -inform der -in MyCertificate.der -out MyCertificate.crt
   ```

Installing the Signed Certificate in Turbonomic

Once you have obtained the signed certificate, you can install it on your Turbonomic instance. You will use the private key and certificate files you obtained when requesting the the signed certificate:

- `myPrivate.key`
- `MyCertificate.crt`

To install the signed certificate:

1. Open an SSH terminal session on your Turbonomic instance.
2. Add the key and certificate data to your Turbonomic charts.yaml file.

   Open the file: `/opt/turbonomic/kubernetes/operator/deploy/crds/charts_v1alpha1_xl_cr.yaml`

   Find the section for `global` parameters. Under the `global` parameters, create the `ingress:secrets` section, and then create entries for `certificate`, `key`, and `name`.

   Your global parameters should be similar to the following:

   ```
   global:
     ingress:
       secrets:
           - certificate: |
               -----BEGIN CERTIFICATE-----
               SAMPLE PUBLIC KEY
               -----END CERTIFICATE-----
             key: |
               -----BEGIN RSA PRIVATE KEY-----
               SAMPLE PRIVATE KEY
               -----END RSA PRIVATE KEY-----
             name: nginx-ingressgateway-certs
   ```

   For the fields you added:

   - `certificate:` This field holds the content of your `MyCertificate.crt` file. Open that file to copy its contents and paste them here.
   - `key:` This field holds the content of your `myPrivate.key` file. Open that file to copy its contents and paste them here.
   - `name:` This field is required, and the name must be `nginx-ingressgateway-certs`.
3. Apply the changes you made to the CR file.

   Execute the command:

   ```
   kubectl apply -f kubernetes/operator/deploy/crds/charts_v1alpha1_xl_cr.yaml
   ```

   This should restart the nginx pod. After restart, Turbonomic will then require a certificate for HTTPS access.

(Optional) Enabling Secure Access for Probes

If your targets require SSL connections via trusted certificate, Turbonomic enables you to install a trusted certificate on the associated probe component.

The Turbonomic platform includes a number of probe components that it uses to connect to targets and discover their data. This procedure assumes setup for one component, the *Dynatrace* probe. You can use the same steps for other probes, providing a different Kubernetes Secret Name for each.

> **NOTE:**
> To configure SSL communication for multiple probes, you must get a unique certificate for each one.

To install a certificate on a probe component, you must know the Kubernetes secret name for the given probe. This table lists the probes that you can configure, plus their secret names:

Probe:	K8s Secret Name:
mediation-awsbilling mediation-aws mediation-awscost mediation-awslambda	aws
mediation-appinsights	appinsights
mediation-newrelic	newrelic
mediation-azuresp mediation-azure mediation-azurevolumes mediation-azurecost	azure
mediation-azureea	azureea
mediation-dynatrace	dynatrace

Installing the Signed Certificate on the Probe Component

This procedure assumes you already have a valid `.crt` file. For steps to get a signed certificate, see Requesting a Certificate *(on page 19)*. However, instead of generating the certificate on your Turbonomic instance, you should generate it on your local machine.

Once you have obtained the signed certificate, you can install it on your probe instance. You will use the certificate file you obtained when requesting the the signed certificate:

```
MyCertificate.crt
```

To install the signed certificate on a probe:

1. Copy the certificate from your local machine to the Turbonomic instance.

   Use SCP to copy the `MyCertificate.crt` from your local machine to the `/tmp` directory on the instance. Once you have done this, you can use `kubectl` to transfer the certificate to the probe component.
2. Open an SSH terminal session on your Turbonomic instance, using the `turbo` user account.

3. Copy the certificate file to the probe component.

   First, get the ID for the pod that runs the probe. To get the ID, execute the command:

   ```
   kubectl get pod
   ```

   This lists the pods running in the Turbonomic platform, including their IDs. Record the ID of the pod you want to configure.

   To copy the certificate, execute the following command, where **<Probe-Pod-Id>** is the ID you recorded:

   ```
   kubectl cp /tmp/MyCertificate.crt <Probe-Pod-Id>:/tmp
   ```

4. Copy the keystore `cacerts` file to the component's `/tmp` directory.

   Use these commands to open a bash session on the pod and copy the file:

   - `kubectl exec -ti <Probe-Pod-Id> bash`
   - `cp /etc/pki/ca-trust/extracted/java/cacerts /tmp`

5. Import the certificate into the pod's keystore.

   As part of this step, you will ensure that the certificate is in Base64 format. While still in the bash session on your probe component, execute the following commands:

   - `chmod 775 /tmp/cacerts`
   - `keytool -import -alias MyCertificate.crt -file /tmp/ca.crt -keystore /tmp/cacerts -deststoretype jks -storepass changeit -no-prompt`

     Where **MyCertificate.crt** is the name of the certificate that you acquired.
   - `base64 /tmp/cacerts > base64.txt`

6. Create the yaml file to contain the certificate data.

   You will create a yaml file using the K8s Secret Name for the probe, with the following naming convention:

   `/tmp/<Secret_Name>-secrets.yaml`

   For example, assume you are enabling SSL for the Dynatrace probe. In that case, the secret name is `dynatrace`, and you would create the yaml file:

   `/tmp/dynatrace-secrets.yaml`

   a. While still in the bash session on the probe component, open the yaml file in a vi editor session:

      `vi /tmp/<Secret_Name>-secrets.yaml`

   b. Then add the following content to the file:

      ```
      apiVersion: v1
      kind: Secret
      metadata:
        name: dynatrace
      data:
        cacerts: |
          xxxxxxxxxxxxxxxxxxxxxxxx
          xxxxxxxxxxxxxxxxxxxxxxxx
      ```

In the `cacerts` section, replace the `xxx` characters with the `base64.txt` data that you generated.

c. Align the base64 data to the yaml format.

Press ESC to exit the edit mode you were in to add the content to the file. Then type `:` to enter the command mode. For the command, type the following, where the whitespace token is four space characters:

```
7,$s/^/    /
```

Press RETURN to execute the command. Then save and exit the vi editor.

7. Exit your session on the probe component.

You now have a yaml file on the probe component that configures a certificate for communications from that probe. You must leave the session on the probe so you can apply the yaml to the Turbonomic platform.

To leave the component session and return to your session on the Turbonomic VM, type `exit` in the shell.

8. Apply the yaml file to the Turbonomic platform.

To apply the yaml file to your platform, you copy it from the probe component to the platform, and then you apply it.

To copy the file from the probe component, you must know the pod ID. To get the ID, execute the command `kubectl get pod`, and record the ID of the pod you have just configured. Then execute the following commands, where **<Probe-Pod-Id>** is the ID you recorded, and :

- `kubectl cp <Probe-Pod-Id>:/tmp/<Secret_Name>-secrets.yaml /tmp/<Secret_Name>-secrets.yaml`
- `kubectl apply -f /tmp/<Secret_Name>-secrets.yaml`

9. For each probe that you configure with a SSL certificate, add an entry in the chart_v1alpha1_cl_cr.yaml file.

a. With a shell session running on the Turbonomic platform, open the following file in a text editor:

```
/opt/turbonomic/kubernetes/operator/deploy/crds/chart_v1alpha1_xl_cr.yaml
```

b. Search the file for the entry for the probe that you are configuring. Use the probe names listed in the table above. For example, if you are configuring the Dynatrace probe, find the entry for `mediation-dynatrace`.

c. Underneath the probe entry, add the following entry for `javaComponentOptions:`

```
javaComponentOptions: -Djavax.net.ssl.trustStore=/etc/targets/cacerts -Dorg.eclipse.jetty.websocket.jsr356.ssl-trust-all=true -DLog4jContextSelector=com.vmturbo.mediation.common.context.ThreadGroupPrefixContextSelector
```

For example, if you are configuring the Dynatrace probe, the entry should be similar to:

```
mediation-dynatrace:
    javaComponentOptions: -Djavax.net.ssl.trustStore=/etc/targets/cacerts -Dorg.eclipse.jetty.websocket.jsr356.ssl-trust-all=true -DLog4jContextSelector=com.vmturbo.mediation.common.context.ThreadGroupPrefixContextSelector
    resources:
      limits:
        memory: 2Gi
```

d. Save and exit the chart_v1alpha1_cl_cr.yaml file.

e. Apply the changed file to your Turbonomic platform.

Execute the command:

```
kubectl apply -f chart_v1alpha1_xl_cr.yaml
```

(Optional) Enabling Embedded Reports

Embedded reporting runs in its own component, as part of the Turbonomic platform. This architecture enhances performance and reduces storage requirements. When you enable Embedded Reports, you can navigate to a set of pre-built dashboards that chart multiple environment details. Dashboards and charts are powered by the Grafana observability platform. With Grafana, it's easy to navigate the existing dashboards, and to make your own charts and dashboards with no coding required.

> **NOTE:**
> We recommend that you do not enable Embedded Reports for environments that contain more than 10,000 VMs.

To enable Embeded Reports, you will:

- Install two containers – Grafana and CURL.
- Install TimescaleDB, a time-series SQL database that provides fast analytics and scalability on a proven storage engine.
- Enable the processes that implement the embedded reporting.
- Update the API pod to enable new search and data ingestion capabilities.
- Double-check the installation

Installing the Grafana and CURL Containers

To run Embedded Reports, you must install the Grafana container that generates the report displays, and the CURL container that manages cmmunications the Embedded Reports require. To download and install these containers:

1. Open a SSH terminal session to your Turbonomic instance.

   Log in with the System Administrator that you set up when you installed Turbonomic:

   - Username: `turbo`
   - Username: `[your_private_password]`
2. Navigate to the /tmp directory.

   Execute `cd /tmp`.
3. Download the two containers.

   Execute the commands to download the containers into `/tmp`:

   - `curl -O http://download.vmturbo.com/appliance/download/updates/7.22.5/grafana.tgz`
   - `curl -O http://download.vmturbo.com/appliance/download/updates/7.22.5/curl.tgz`
4. Optionally, validate the checksum hash.

The commands and results should be:

- `sha256sum /tmp/grafana.tgz`

  ```
  08302e7c681efd39c9bcedc9349b68dc637399e73217b6a12835227d99ad26bc  /tmp/grafa
  na.tgz
  ```

- `sha256sum curl.tgz`

  ```
  0d09d752bbf676c7f6d83dd2050999a1d6053b430e2e3be39ad272cad21b9157  curl.tgz
  ```

5. Load the containers into your environment.

   Excute the commands:

   - `sudo docker load -i /tmp/grafana.tgz`
   - `sudo docker load -i /tmp/curl.tgz`

Installing TimescaleDB

The instructions to install TimescaleDB are different, depending on the current status of your Turbonomic instance:

Turbonomic Status	Installation Tasks
New .OVA You have installed Turbonomic version 7.22.5 or later as a new .OVA (see Installing on VMware Systems *(on page 7)*).	TimescaleDB is fully installed. You can skip to *Enabling the Embedded Reports Feature*.
Offline Update You have updated from an earlier version to 7.22.5 or later, and used the Offline Update procedure (see Offline Update *(on page 42)*).	TimescaleDB is installed but not configured. You must execute the two configuration scripts. The offline update loads these scripts onto your Turbonomic instance.
Online Update You have updated from an earlier version to 7.22.5 or later, and used the Online Update procedure (see Online Update *(on page 43)*).	TimescaleDB is *not* installed or configured, and the scripts are not in place on your Turbonomic instance. You must download all of the scripts, and you must execute them.

> **NOTE:**
> It is important that you understand which Turbonomic Status applies to your instance. If TimescaleDB is already installed, you *must not* run the installation script again.

To install TimescaleDB (for Turbonomic updates, only – not for a new installation of Turbonomic):

1. Open a SSH terminal session to your Turbonomic instance.

   Log in with the System Administrator that you set up when you installed Turbonomic:

   - Username: `turbo`
   - Username: `[your_private_password]`

2. If your Turbonomic status is Online Update, install the required scripts.

If you are performing an *Online Update*, then you must install the scripts, and the bc package.

a. Get the following three scripts:

- `install_timescaledb.sh`

  https://raw.githubusercontent.com/turbonomic/t8c-install/7.22/bin/install_timescaledb.sh
- `configure_timescaledb.sh`

  https://raw.githubusercontent.com/turbonomic/t8c-install/7.22/bin/configure_timescaledb.sh
- `switch_dbs_mount_point.sh`

  https://raw.githubusercontent.com/turbonomic/t8c-install/7.22/bin/switch_dbs_mount_point.sh

b. Save these three scripts on the Turbonomic server.

Save each script to `/opt/local/bin`:

- `/opt/local/bin/install_timescaledb.sh`
- `/opt/local/bin/configure_timescaledb.sh`
- `/opt/local/bin/switch_dbs_mount_point.sh`

c. Install the bc package.

This package supports some of the script commands. Execute the command, `yum install -y bc`.

3. Run the following scripts:

- For an *Offline Update*, run:

  a. `/opt/local/bin/configure_timescaledb.sh`

  b. `/opt/local/bin/switch_dbs_mount_point.sh`

- For an *Online Update*, run:

  a. `/opt/local/bin/install_timescaledb.sh`

  b. `/opt/local/bin/configure_timescaledb.sh`

  c. `/opt/local/bin/switch_dbs_mount_point.sh`

Enabling the Embedded Reports Feature

You must enable the Grafana Exporter, TimescaleDB, and data extraction processes. To do this, edit the `charts_v1alpha1_xl_cr.yaml` file.

1. While still in the SSH session on your Turbonomic instance, open the following file in a text editor:

   `/opt/turbonomic/kubernetes/operator/deploy/crds/charts_v1alpha1_xl_cr.yaml`

2. Specify the IP address of the Turbonomic instance for external access to the TimescaleDB database.

   In the `global:` section of the file, add the following line, where **`<Platform_IP>`** is the IP address of your instance:

   ```
   global:
     externalTimescaleDBIP: <Platform_IP>
   ```

3. Enable the Grafana process.

   Find the `grafana:` section in the `crds/charts_v1alpha1_xl_cr.yaml` file, and uncomment the line, `enabled: true`.

4. Enable Postgres as the database type.

   Enabling Postgres sets persistent storage of historical data for Embedded Reports. In the `grafana:` section, find the subsection for `grafana.ini: database:` and uncomment the line, `type: postgres`.

   The changes you have made so far should be similar to:

   ```
   global:
       externalTimescaleDBIP: <Platform_IP>
       ...

   grafana:
       enabled: true
       adminPassword: admin
       grafana.ini:
         database:
           type: postgres
           ...
   ```

5. Change the admin and database passwords.

   It is good practice to change any passwords, and not keep their default values. To set the passwords for Embedded Reports, you will change two that are already set in the .cr file, and add another one.

   > **IMPORTANT:**
   > *Use only alpha-numeric characters for these passwords.*
   >
   > Thee passwords enable communication between the various Embedded Reports components. Some of the components only accept alpha-numeric characters. If you use special characters, then the components will not be able to communicate. Further, the steps to correct these passwords require assistance from your Support engineer.

   To set the passwords:

   - Set the Grafana admin password.

     This password enable access to Grafana from external URLs and also from the extractor component that feeds data to Grafana. In the `grafana:` section, change the value of `adminPassword`.

     Do not use special characters.

     Assume your password is `MyNewGrafanaPassword`. Then you would set `adminPassword:` **`MyNewGrafanaPassword`**

   - Set the Grafana database password.

This password enables communication between Grafana and the Postgres database that stores the reporting data. In the `grafana:` section, find the subsection for `grafana.ini: database: password:` and change the password value.

- Add a Properties section, and specify the password that the extractor component will use to communicate with Grafana.

> **IMPORTANT:**
> This password *must match* the `grafana: adminPassword:` that you set above. Do not use special characters.

Assume your password is `MyNewGrafanaPassword`. Just after the `grafana:` section, and at the same level to it, add the following entry:

```
properties:
  extractor:
      grafanaAdminPassword: MyNewGrafanaPassword
```

It is important that you align this entry with the indentation for the `grafana:` entry. The changes you have made so far should be similar to the following. Notice that you use the same password for the Grafana adminPassword, and for the Extractor grafanaAdminPassword. Also notice that you must use only alphanumeric characters for the passwords:

```
global:
    externalTimescaleDBIP: <Platform_IP>
    ...

grafana:
    enabled: true
    adminPassword: MyNewGrafanaPassword
    grafana.ini:
      database:
        type: postgres
        password: MyNewDatabasePassword

properties:
    extractor:
        grafanaAdminPassword: MyNewGrafanaPassword
```

6. Enable the three Embedded Reports processes.

   Just after the `properties:` section that you added, and at the same level to it, add the following entries to enable the reporting processes:

```
reporting:
  enabled: true
timescaledb:
  enabled: true
extractor:
  enabled: true
```

   It is important that you align these entries with the indentation for the `grafana:` section and the `properties:` section. The changes you have made should now be similar to:

```
global:
```

```
    externalTimescaleDBIP: <Platform_IP>
    ...

grafana:
    enabled: true
    adminPassword: MyNewGrafanaPassword
    grafana.ini:
      database:
        type: postgres
        password: MyNewDatabasePassword

properties:
    extractor:
        grafanaAdminPassword: MyNewGrafanaPassword

reporting:
    enabled: true
timescaledb:
    enabled: true
extractor:
    enabled: true
```

7. When you are done editing the `charts_v1alpha1_xl_cr.yaml` file, save and apply your changes.
   - Save your changes and quit the text editor.
   - Apply the changes.

     Execute the command:

     ```
     kubectl apply -f /opt/turbonomic/kubernetes/operator/deploy/crds/charts_v1alp
     ha1_xl_cr.yaml
     ```

   - Delete the api and extractor pods.

     Deleting these pods triggers them to restart, which loads the changes you made.

     To get the full pod names, execute the command, `kubectl get pods -n turbonomic`. Then find the two entries for the pods that begin with `api` and `extractor`. For example, assume the entries are:

     ```
     ...
     api-7887c66f4b-shndq                          1/1     Running   0
     ...
     extractor-5b86976bc8-vxwz4                    1/1     Running   0
     ...
     ```

     Then you would execute the commands:

     - `kubectl delete pod -n turbonomic api-7887c66f4b-shndq`
     - `kubectl delete pod -n turbonomic extractor-5b86976bc8-vxwz4`

Double-checking the Installation

To double-check the installation:

- Verify that the Embedded Reports pods are running.

  To verify that the pods are running, execute `kubectl get pods -n turbonomic`. The output should include entries similar to:

```
NAME                                        READY   STATUS    RESTARTS
extractor-7759dbcb47-vs6hr                  1/1     Running   0
grafana-84ccb4bfb-l7sp7                     1/1     Running   0
```

- Verify that Postgres is running.

  The Postgres database should be running as a daemon on the Turbonomic server machine. To check the status, execute the command:

  `sudo systemctl status postgresql-12.service`.

  You should see output similar to:

```
postgresql-12.service - PostgreSQL 12 database server
Loaded: loaded (/usr/lib/systemd/system/postgresql-12.service; enabled; vendor prese
t: disabled)
Active: active (running) since Wed 2020-07-29 06:39:43 UTC; 14h ago
     Docs: https://www.postgresql.org/docs/12/static/
Process: 1536 ExecStartPre=/usr/pgsql-12/bin/postgresql-12-check-db-dir ${PGDATA} (c
ode=exited, status=0/SUCCESS)
Main PID: 1562 (postmaster)
    Tasks: 15
   Memory: 145.5M
   CGroup: /system.slice/postgresql-12.service
           ##  419 postgres: TimescaleDB Background Worker Scheduler
           ## 1562 /usr/pgsql-12/bin/postmaster -D /var/lib/pgsql/12/data/
           ## 1928 postgres: logger
           ## 1986 postgres: checkpointer
           ## 1988 postgres: background writer
           ## 1989 postgres: walwriter
           ## 1990 postgres: autovacuum launcher
           ## 1991 postgres: stats collector
           ## 1992 postgres: TimescaleDB Background Worker Launcher
           ## 1994 postgres: logical replication launcher
           ## 4054 postgres: grafana_backend grafana 10.233.90.172(33038) idle
           ## 4884 postgres: grafana_backend grafana 10.233.90.172(35814) idle
           ## 4912 postgres: grafana_reader extractor 10.233.90.172(33898) idle
           ##11365 postgres: grafana_reader extractor 10.233.90.172(40728) idle
           ##32367 postgres: TimescaleDB Background Worker Scheduler
```

(Optional) Changing the IP Address of the Platform Node

For standard installations of Turbonomic, you might need to move the platform from one location to another, and that move can require a new IP address. If you must change the IP address of the platform, you can use the supplied script.

To make such a change:

1. Get your information ready.

Identify the location changes you need to make, and identify both the current IP address for your platform, and the new IP address you will use.

You must also know the credentials to open a shell session on the VM and run commands.

2. Shut down the platform VM.

   Use your hypervisor management platform to shut down the VM that hosts your Turbonomic platform.

3. Create a full snapshot of the VM.

   It is important to make a full snapshot of your installation before you try to move it or modify its IP address.

4. Move the VM to its new location.

   Use your Hypervisor management platform to relocate the VM, assign it a new IP address, and restart it.

5. Run the script to propagate the new IP address across the Turbonomic components.

   a. Open a shell session on the newly located VM.

   b. If your Turbonomic server is earlier than 7.22.4, install the script on the server.

      If you installed Turbonomic a version earlier than 7.22.4, and you have updated to 7.22.4, you need to install the script. In the shell session, execute the following commands:

      i. `cd /opt/local/bin`

      ii. `curl -0 https://raw.githubusercontent.com/turbonomic/t8c-install/master/bin/kubeNodeIPChange.sh`

   c. Execute the script to change the IP addresses.

      `sudo /opt/local/bin/kubeNodeIPChange.sh`

      The script discovers the old IP address from the original CR configuration for the Kubernetes node, and it discovers the new IP address via the `ifconfig eth0` command.

   d. Verify that the storage pods are running.

      Execute the command:

      `kubectl get pods -n default`

      The output should include the following for the storage pods, where **{UUID}** is a unique ID for the pod name:

      ```
      NAME              READY   STATUS    RESTARTS   AGE
      glusterfs-{UUID}  1/1     Running   1          6d2h
      heketi-{UUID}     1/1     Running   7          6d2h
      ```

      The READY state should be `1/1` and the STATUS should be `RUNNING`. If either of the pods do not show these states, then restart both pods and wait for them to come up. This should take approximately five minutes.

   e. Verify that the change is successful.

      Log into the Turbonomic user interface for the newly located installation, and ensure that it displays correctly. You should review the Supply Chain, your groups, and your policies. You should also ensure that charts show data correctly.

      When you are sure that the change is successful, you can remove the snapshot you made of the VM in its old location.

License Installation and First-time Login

Before you begin, make sure you have your full or trial license key file that was sent to you in a separate email. Save the license file on your local machine so you can upload it to your Turbonomic installation.

To use Turbonomic for the first time, perform the following steps:

1. Type the IP address of your installed Turbonomic instance in a Web browser to connect to it.
2. Log in to Turbonomic.
   - Use the default credential for **USERNAME**: `administrator`.
   - Type a password for **PASSWORD**.
   - Type the password again to verify it for **REPEAT PASSWORD**.
   - Click **CONFIGURE**.
3. Continue setting up your Turbonomic installation.

   Click **LET'S GO**.
4. Open the **Enter License** fly-out.

   Click **IMPORT LICENSE**.
5. Upload your license key file.
   a. In the Enter License fly-out, you can upload the license in one of the following ways:
      - Drag the license key file into the Enter License fly-out.
      - Browse to the license key file.

      Be sure to upload only .xml or .lic files.
   b. Click **SAVE**.

Depending on which license you have installed, the license enables either a trial or a full unlimited license for Turbonomic.

Single Sign-On Authentication

If your company policy supports Single Sign-On (SSO) authentication, Turbonomic enables SSO authentication by using Security Assertion Markup Language (SAML) 2.0.

At a high-level, the process involves:

- Creating external groups or at least one external user for SSO. See "Managing User Accounts" in the *Turbonomic User Guide*.
- Configuring Turbonomic to connect to the SAML Identity Provider (IdP). See Configuring Single Sign-On *(on page 35)*.

When SSO is enabled, use your SSO credentials to log in to your Turbonomic instance. Do not use your local or Active Directory (AD) credentials for the login. The Identity Provider (IdP) will perform the authentication.

Prerequisites

Before you begin, make sure the IdP is set up for SSO. You can use a proprietary or public IdP. For examples of settings for a public Okta IdP, see What Are the Typical Settings for an IdP? *(on page 50)*.

Configuring Single Sign-On

To configure Single Sign-On, perform these steps:

1. (Required) Create external groups or at least one external user for SSO.

   > **IMPORTANT:**
   > When SSO is enabled, Turbonomic only permits logins via the SSO IdP. Whenever you navigate to your Turbonomic installation, it redirects you to the SSO Identity Provider (IdP) for authentication before displaying the Turbonomic user interface.
   >
   > Before you enable SSO for your Turbonomic installation, *you must configure at least one SSO user with Turbonomic administrator privileges*. If you do not, then once you enable SSO you will not be able to configure any SSO users in Turbonomic. To authorize an SSO user as an administrator, use **EXTERNAL AUTHENTICATION** to do one of the following:
   >
   > - Configure a single SSO user with administrator authorization.
   >
   >   Add an external user. The username must match an account that is managed by the IdP.
   > - Configure an SSO user group with administrator authorization.
   >
   >   Add an external group. The group name must match a user group on the IdP, and that group must have at least one member.
   >
   > For information about creating external groups or external users for SSO, see "Managing User Accounts" in the *Turbonomic User Guide*.

2. (Required) Ensure that chrony is configured and the system time on your Turbonomic instance is correct.

   For instructions, see Synchronizing Time *(on page 15)*.

3. Obtain the metadata from your IdP.

   You will use this metadata to configure SSO in the CR file located at:

   ```
   /opt/turbonomic/kubernetes/operator/deploy/crds/charts_v1alpha1_xl_cr.yaml
   ```

   To get the metadata:

   a. Contact your security administrator to obtain the metadata from IdP.
   b. Save the metadata file in a directory on your local machine. For example, save the file to:

      ```
      /tmp/MySamlMetadata.txt
      ```

   c. Compare your metadata to the sample provided in Example of IdP Metadata *(on page 38)*.

      Cat out the file you just saved. It should be similar to the provided sample.

4. Obtain a certificate from IdP.

   Contact your security administrator to obtain a certificate from IdP.

5. Update the CR file with your SAML configuration.

   You now have the data that you need to configure SSO via SAML. You will edit the `cr.yaml` file that configures your Turbonomic node, and then deploy or restart the node.

   - Display the contents of your downloaded SAML metadata.

     For example, assuming you saved the file to this location on your local machine, execute the command:

     ```
     cat /tmp/MySamlMetadata.txt
     ```

- Open the CR file for editing.

  In a shell, cd to the deploy/crds directory in the Turbonomic VM:

  ```
  cd /opt/turbonomic/kubernetes/operator/deploy/crds
  ```

  Then open the CR file for editing. For example, to open the file in VI:

  ```
  vi charts_v1alpha1_xl_cr.yaml
  ```

  As you edit this file, you will refer to the metadata that you obtained from yor IdP.
- In the CR file, navigate to the entry for the API component.

  In the CR file search for or scroll to the entry:

  ```
  apiVersion: charts.helm.k8s.io/v1alpha1
  ```

  You will make changes to this component spec, under `spec:properties:api:`
- Turn on the SAML feature.

  For the first API property, set the following:

  ```
  samlEnabled: true
  ```
- Set the SSO endpoint

  In the SAML metadata, find the entry for `md:SingleSignOnService`. Within that element, find the `Location` attribute. The value of `Location` is the SSO endpoint. Using the sample metadata we have provided, you would make the following setting in your CR file:

  ```
  samlWebSsoEndpoint: https://dev-771202.oktapreview.com/app/ibmdev771202_turbo2_1/
  exkexl6xc9MhzqiC30h7/sso/saml
  ```
- Set the SAML entity ID

  In the SAML metadata, find the entry for `md:EntityDescriptor`. Within that element, find the `entityID` attribute. Using the sample metadata we have provided, you would make the following setting in your CR file:

  ```
  samlEntityId: http://www.okta.com/exkexl6xc9MhzqiC30h7
  ```
- Set the SAML registration

  Set the following property:

  ```
  samlRegistrationId: simplesamlphp
  ```
- Set the SAML SP entity ID

  Set the following property:

  ```
  samlSpEntityId: turbo
  ```
- Enter the SAML certificate

  In the metadata that you got from your IdP, find the entry for `<ds:X509Certificate>`. Copy the contents of this tag – copy the characters that are between `<ds:X509Certificate>` and `</ds:X509Certificate>`.

  Create an entry for the certificate in the API properties section of the CR file. On a new line, enter:

  ```
  samlIdpCertificate: |
  ```

Then open a new line after the entry you just created, and paste the certificate content that you copied from your metadata file.

The finished API section of the CR file should be similar to the following:

```
apiVersion: charts.helm.k8s.io/v1alpha1
kind: Xl
metadata:
  name: xl-release
spec:
  properties:
    api:
      samlEnabled: true
      samlWebSsoEndpoint: https://dev-771202.oktapreview.com/app/ibmdev771202_tur
bo2_1/exkexl6xc9MhzqiC30h7/sso/saml
      samlEntityId: http://www.okta.com/exkfdsn6oy5xywqCO0h7
      samlRegistrationId: simplesamlphp
      samlSpEntityId: turbo
      samlIdpCertificate: |
        MIIDpDCCAoygAwIBAgIGAWMnhv7cMA0GCSqGSIb3DQEBCwUAMIGSMQswCQYDVQQGEwJVUzETMBEG
        A1UECAwKQ2FsaWZvcm5pYTEWMBQGA1UEBwwNU2FuIEZyYW5jaXNjbzENMAsGA1UECgwET2t0YTEU
        MBIGA1UECwwLU1NPUHJvdmlkZXIxEzARBgNVBAMMCmRldi03NzEyMDIxHDAaBgkqhkiG9w0BCQEW
        DWluZm9Ab2t0YS5jb20wHhcNMTgwNTAzMTk0MTI4WhcNMjgwNTAzMTk0MjI4WjCBkjELMAkGA1UE
        BhMCVVMxEzARBgNVBAgMCkNhbGlmb3JuaWExFjAUBgNVBAcMDVNhbiBGcmFuY2lzY28xDTALBgNV
        BAoMBE9rdGExFDASBgNVBAsMC1NTT1Byb3ZpZGVyMRMwEQYDVQQDDApkZXYtNzcxMjAyMRwwGgYJ
        KoZIhvcNAQkBFg1pbmZvQG9rdGEuY29tMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA
        ugxQGqHAXpjVQZwsO9n8l8bFCoEevH3AZbz7568XuQm6MK6h7/O9wB4C5oUYddemt5t2Kc8GRhf3
        BDXX5MVZ8G9AUpG1MSqe1CLV2J96rMnwMIJsKeRXr01LYxv/J4kjnktpOC389wmcy2fE4RbPoJne
        P4u2b32c2/V7xsJ7UEjPPSD4i8l2QG6qsUkkx3AyNsjo89PekMfm+Iu/dFKXkdjwXZXPxaL0HrNW
        PTpzek8NS5M5rvF8yaD+eE1zS0I/HicHbPOVvLal0JZyN/f4bp0XJkxZJz6jF5DvBkwIs8/Lz5GK
        nn4XW9Cqjk3equSCJPo5o1Msj8vlLrJYVarqhwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQC26kYe
        LgqjIkF5rvxB2QzTgcd0LVzXOuiVVTZr8Sh57l4jJqbDoIgvaQQrxRSQzD/X+hcmhuwdp9s8zPHS
        JagtUJXiypwNtrzbf6M7ltrWB9sdNrqc99d1gOVRr0Kt5pLTaLe5kkq7dRaQoOIVIJhX9wgynaAK
        HF/SL3mHUytjXggs88AAQa8JH9hEpwG2srN8EsizX6xwQ/p92hM2oLvK5CSMwTx4VBuGod70EOwp
        6Ta1uRLQh6jCCOCWRuZbbz2T3/sOX+sibC4rLIlwfyTkcUopF/bTSdWwknoRskK4dBekFcvN9N+C
        p/qaHYcQd6i2vyor888DLHDPXhSKWhpG
```

6. Save your changes to the CR file.
7. Restart the API component to load the new spec.
   a. Open an SSH terminal session to your Turbonomic instance.
   b. Use sudo as root.

      ```
      sudo bash
      ```

   c. Restart the API component.

      ```
      turboctl restart api
      ```

8. Verify that the configuration is successful.
   a. Navigate to the Turbonomic User Interface.

      You will be automatically redirected to your IdP for authentication.

   b. Log in with the username that is a member of the external group or external user previously configured.
   c. Verify that the system time on your Turbonomic instance is correct.

If the time is not synchronized, this might cause an `HTTP Status 401 -authentication failed` exception in the browser.

d. If the configuration is not successful, look for an `HTTP Status 500` exception in the product log. If this exception exists, review your CR file for invalid entries.

Example of IdP Metadata

This section provides an example of IdP metadata which may be useful when you are examining the optional attributes in your metadata.

If your metadata includes optional attribute tags that are not listed in the example, remove those optional attribute tags since they are not supported.

```
<?xml version="1.0" encoding="UTF-8"?>
     <md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
     entityID="http://www.okta.com/exkexl6xc9MhzqiC30h7">
     <md:IDPSSODescriptor WantAuthnRequestsSigned="false"
     protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
     <md:KeyDescriptor use="signing">
     <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
     <ds:X509Data>
     <ds:X509Certificate>
     MIIDpDCCAoygAwIBAgIGAWMnhv7cMA0GCSqGSIb3DQEBCwUAMIGSMQswCQYDVQQGEwJVUzETMBEG
     A1UECAwKQ2FsaWZvcm5pYTEWMBQGA1UEBwwNU2FuIEZyYW5jaXNjbzENMAsGA1UECgwET2t0YTEU
     MBIGA1UECwwLU1NPUHJvdmlkZXIxEzARBgNVBAMMCmRldi03NzEyMDIxHDAaBgkqhkiG9w0BCQEW
     DWluZm9Ab2t0YS5jb20wHhcNMTgwNTAzMTk0MTI4WhcNMjgwNTAzMTk0MjI4WjCBkjELMAkGA1UE
     BhMCVVMxEzARBgNVBAgMCkNhbGlmb3JuaWExFjAUBgNVBAcMDVNhbiBGcmFuY2lzY28xDTALBgNV
     BAoMBE9rdGExFDASBgNVBAsMC1NTT1Byb3ZpZGVyMRMwEQYDVQQDDApkZXYtNzcxMjAyMRwwGgYJ
     KoZIhvcNAQkBFg1pbmZvQG9rdGEuY29tMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA
     ugxQGqHAXpjVQZwsO9n8l8bFCoEevH3AZbz7568XuQm6MK6h7/O9wB4C5oUYddemt5t2Kc8GRhf3
     BDXX5MVZ8G9AUpG1MSqe1CLV2J96rMnwMIJsKeRXr01LYxv/J4kjnktpOC389wmcy2fE4RbPoJne
     P4u2b32c2/V7xsJ7UEjPPSD4i8l2QG6qsUkkx3AyNsjo89PekMfm+Iu/dFKXkdjwXZXPxaL0HrNW
     PTpzek8NS5M5rvF8yaD+eE1zS0I/HicHbPOVvLal0JZyN/f4bp0XJkxZJz6jF5DvBkwIs8/Lz5GK
     nn4XW9Cqjk3equSCJPo5o1Msj8vlLrJYVarqhwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQC26kYe
     LgqjIkF5rvxB2QzTgcd0LVzXOuiVVTZr8Sh57l4jJqbDoIgvaQQrxRSQzD/X+hcmhuwdp9s8zPHS
     JagtUJXiypwNtrzbf6M7ltrWB9sdNrqc99d1gOVRr0Kt5pLTaLe5kkq7dRaQoOIVIJhX9wgynaAK
     HF/SL3mHUytjXggs88AAQa8JH9hEpwG2srN8EsizX6xwQ/p92hM2oLvK5CSMwTx4VBuGod70EOwp
     6Ta1uRLQh6jCCOCWRuZbbz2T3/sOX+sibC4rLIlwfyTkcUopF/bTSdWwknoRskK4dBekFcvN9N+C
     p/qaHYcQd6i2vyor888DLHDPXhSKWhpG
     </ds:X509Certificate>
     </ds:X509Data>
     </ds:KeyInfo>
     </md:KeyDescriptor>
     <md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</md:Nam
eIDFormat>
     <md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</md:N
ameIDFormat>
     <md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POS
T"
     Location="https://dev-771202.oktapreview.com/app/ibmdev771202_turbo2_1/exkex
l6xc9MhzqiC30h7/sso/saml"/>
```

```
        <md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Red
irect"
        Location="https://dev-771202.oktapreview.com/app/ibmdev771202_turbo2_1/exkex
l6xc9MhzqiC30h7/sso/saml"/>
        </md:IDPSSODescriptor>
        </md:EntityDescriptor>
```

Disabling Single Sign-On

If for some reason you no longer want to use SSO, you can disable it for your Turbonomic installation. To disable Single Sign-On, perform these steps:

1. Update the SAML configuration to disable it.
    a. Open an SSH terminal session to your Turbonomic instance.
    b. Open the CR file for editing.

        In a shell, cd to the deploy/crds directory in the Turbonomic VM:

        ```
        cd /opt/turbonomic/kubernetes/operator/deploy/crds
        ```

        Then open the CR file for editing. For example, to open the file in VI:

        ```
        vi charts_v1alpha1_xl_cr.yaml
        ```

    c. In the CR file, navigate to the entry for the API component.

        In the CR file search for or scroll to the entry:

        ```
        apiVersion: charts.helm.k8s.io/v1alpha1
        ```

        You will make changes to this component spec, under `spec:properties:api:`
    d. Turn off the SAML feature.

        Find the `samlEnabled:` property to `false`. It should appear as follows:

        ```
        samlEnabled: false
        ```

    e. Save your changes to the CR file.
2. Restart the API component.

    In the same SSH terminal session that you opened to edit the CR file:

    a. Use sudo as root.

        ```
        sudo bash
        ```

    b. Restart your API component.

        ```
        turboctl restart api
        ```

3. Verify that the configuration is successful.
    a. Navigate to the Turbonomic User Interface.

You will no longer be redirected to your IdP for authentication. You will be redirected to the default Turbonomic login screen.

b. Log in with a local account or an Active Directory (AD) account.

Updating Turbonomic to a New Version

NOTE:
Before you update your Turbonomic installation, please review these issues:

Single Sign-On Authentication	*For upgrades from versions earlier than 7.22.0*: Starting with 7.22.0, Turbonomic supports SAML 2.0. To use this version of authentication, you must reconfigure your SAML support. See Single Sign-On Authentication *(on page 34)*.
Minimum Requirements	*For upgrades from versions earlier than 7.21.4*: Starting with version 7.21.4, the minimum requirements have changed. If you are updating from an earlier version, please review the minimum requirements to determine whether you should inclrease the memory or CPU for your Turbonomic server. See Minimum Requirements *(on page 5)*.
Restoring Historical Data	*For upgrades from versions earlier than 7.21.4*: To optimize the management of historical data in Turbonomic, we have changed the historical database in ways that affect your access to the data. When you upgrade, you will lose access to historical data for a number of entity types. However, if this data is important to you, you can run a script to migrate it to the new database structure. See Restoring Historical Data After an Upgrade *(on page 45)*.

Turbonomic continually and rapidly innovates and improves all aspects of this product. This means that Turbonomic periodically releases newer versions of this product. You should check regularly to see if a new version is available.

When a new version is available, it is important to properly update your existing installed instance, rather than just install a new one. When you first installed Turbonomic, you put into place sophisticated data collection and analysis processes. Internal to the installation is an integrated database that retains performance data from across your virtual environment. Turbonomic uses this historical data for right-sizing, projecting trends, and other analysis. This means

that the database is important to Turbonomic *and becomes more so over time*. Properly updating your installation of Turbonomic preserves the database for continued use.

Before you begin the update procedure:

- Make sure you have the email that Turbonomic sent to you with links to the Turbonomic `.0VA` file and to the ISO image.
- Ensure that the physical machine hosting the VM meets the minimum requirements (see Minimum Requirements *(on page 5)*).

You can update your Turbonomic VM using the offline method or the online method if you have access to the Internet.

Offline Update

To perform an offline update of your Turbonomic installation:

1. Save a snapshot of your current Turbonomic VM.

   Before updating, you should properly shut down (not power off) the Turbonomic VM. To do so, type:

   ```
   sudo init 0
   ```

   Then, perform a snapshot (or clone the VM). This provides a reliable restore point you can turn to in the event that trouble occurs during the update. After you have the snapshot, bring the VM back online.
2. Download and attach the ISO image to the VM that runs Turbonomic.

   Refer to the email you received from Turbonomic for links to the Turbonomic `.0VA` file and to the ISO image.
3. Mount the ISO image by logging in to vCenter.
   a. In vCenter, navigate to the Turbonomic VM.
   b. Right-click the VM and choose **Edit Settings**.
   c. In the CD/DVD Drive drop-down menu:
      i. Choose **Datastore ISO**.
      ii. Browse to the Turbonomic update ISO image and choose it.
   d. Ensure that the **Connect at power on** checkbox is selected.
4. Log in to the Turbonomic VM.

   Use SSH to log in to the Turbonomic VM using the turbo account and password.
5. Make the directory and mount the ISO image.

   Type:

   ```
   sudo su -
   mkdir /mnt/iso
   mount /dev/cdrom /mnt/iso
   ```

6. Verify the correct version of the ISO image is mounted.

   Type: `ls /mnt/iso`

   Verify that the ISO image contains the correct version for your update.
7. Load the latest Docker images.

Type: `/mnt/iso/turboload.sh`

8. Execute these commands to update Turbonomic.

   ```
   /mnt/iso/turboupgrade.sh | tee /opt/turbonomic/t8c_upgrade_$(date +%Y-%m-%d_%H_%M_%S).log
   ```

   Wait until the script is finished.

9. Clear your browser data and refresh your browser.

   After clearing the browser data and refreshing your browser, you have full access to Turbonomic features. However, features that rely on current analysis data will not be available until after a full market cycle — usually 10 minutes. For example, the Pending Actions charts will not show any actions until after a full market cycle.

10. Notify other users to clear their browser data and refresh their Turbonomic browser sessions.

Online Update

This method assumes that you have direct access to the Internet or access to the Internet through a proxy server.

> **NOTE:**
> If you are installing from behind a firewall, make sure you can access `https://index.docker.io` through ports `443` and `80`.

To perform an online update of your Turbonomic installation:

1. Save a snapshot of your current Turbonomic VM.

   Before updating, you should properly shut down (not power off) the Turbonomic VM. To do so, type:

   ```
   sudo init 0
   ```

   Then, perform a snapshot (or clone the VM). This provides a reliable restore point you can turn to in the event that trouble occurs during the update. After you have the snapshot, bring the VM back online.

2. Update your installation of operator.

   If you are updating from a version earlier than 7.22.4, then you must update operator to declare cluster roles.

   If you are updating from an earlier release family, then you must also update operator to recognize that new family. You identify release families by the first two point tokens in the version number. For example, `7.21.3` and `7.22.4` are from different release families; `7.21` and `7.22`, respectively.

   To update operator, execute the following commands in your shell session:

   a. Navigate to the location of the `operator.yaml` file:

      ```
      cd /opt/turbonomic/kubernetes/operator/deploy/
      ```

   b. If you are updating from a version earlier than 7.22.4, update the roles where **`{Release_Family}`** is the release family you are updating to. You will download new yaml files to define cluster role binding, delete the old role bindings from your namespace, and then apply the new role bindings from the yaml files that you downloaded.

      Execute the following commands:

      - `curl -O https://raw.githubusercontent.com/turbonomic/t8c-install/`**`{Release_Family}`**`/operator/deploy/cluster_role.yaml`

- `curl -O https://raw.githubusercontent.com/turbonomic/t8c-install/{Release_Family}/operator/deploy/cluster_role_binding.yaml`
- `kubectl delete -f role.yaml -n turbonomic`
- `kubectl delete -f role_binding.yaml -n turbonomic`
- `kubectl apply -f cluster_role.yaml -n turbonomic`
- `kubectl apply -f cluster_role_binding.yaml -n turbonomic`

c. If you are updating to a new release family, update the family, where **{Release_Family}** is the release family you are updating to.

Execute the following commands:

- First back up the current operator file:

  `mv operator.yaml operator.bak`
- `curl -O https://raw.githubusercontent.com/turbonomic/t8c-install/{Release_Family}/operator/deploy/operator.yaml`
- `kubectl apply -f operator.yaml -n turbonomic`

3. Update the version in your charts_v1alpha1_xl_cr.yaml file.

Open `charts_v1alpha1_xl_cr.yaml` for editing. For example:

```
vi /opt/turbonomic/kubernetes/operator/deploy/crds/charts_v1alpha1_xl_cr.yaml
```

Then edit the file to specify which Turbonomic version that you are installing. In the `global` section, edit the value for the tag parameter. For example, for an update to 7.22.4, the changed entry would appear as follows:

```
global:
#  registry: index.docker.io
#  imageUsername: turbouser
#  imagePassword: turbopassword
  repository: turbonomic
  tag: 7.22.4
```

After you make your change, save the file.

4. Start the online update.

Execute the kubectl apply command:

```
kubectl apply -f
/opt/turbonomic/kubernetes/operator/deploy/crds/charts_v1alpha1_xl_cr.yaml
```

This command downloads the newer Docker images from the Docker Hub and then upgrades the components.

Wait until the update is finished. You should see the message:

```
XL upgrade finished successfully
```

5. Clear your browser data and refresh your browser.

After clearing the browser data and refreshing your browser, you have full access to Turbonomic features. However, features that rely on current analysis data will not be available until after a full market cycle — usually 10 minutes. For example, the Pending Actions charts will not show any actions until after a full market cycle.

6. Notify other users to clear their browser data and refresh their Turbonomic browser sessions.

Restoring Historical Data After an Upgrade

For upgrades from versions earlier than 7.21.4:

To optimize the management of historical data in Turbonomic, we have changed the historical database in ways that affect your access to the data. When you upgrade, you will lose access to historical data for a number of entity types. However, if this data is important to you, you can run a script to migrate it to the new database structure. The upgrade does not automatically migrate this data for the following reasons:

- You must shut down the Turbonomic platform to perform this migration
- For large environments, the migration can take many hours
- Loss of this data does not adversely affect Turbonomic analysis, action generation, or performance maintenance in your environment

The affected historical data is for the following types of entities:

- Application Server
- Virtual Application
- Database Server
- Database
- Load Balancer
- Business Application

If you do not migrate this historical data, then charts and reports for these entities will begin their history from the day you update your installation of Turbonomic. Also, any analysis that relies on historical data (for example, analysis that uses an observation period to calculate percentiles) must begin collecting historical data from the time of your update. In other respects, your installation of Turbonomic should not be affected.

Note that the historical data remains on your database for an amount of time that is determined by your data retention policies. For this reason, you can migrate the data *after* you have performed the product update.

Requirements

- Access to the data migration script

  Download the script from the following location:

  https://github.com/turbonomic/t8c-install/blob/master/bin/convertMaxUtilizationToSettingOverride.py

- Python and the mysql-connector-python module

  To execute the data migration script on your Turbonomic platform, you must install Python and the mysql-connector-python module. To install these on a default Turbonomic OVA, execute the following commands:

  - `sudo yum install python-pip`
  - `pip install mysql-connector-python --user`
  - If you require SSL to connect with the database, then you must use Python 3, and you must install pyopenssl:

    `pip install pyopenssl`

Preparing to Run the Migration Script

Before you execute the script, you must shut down all the Turbonomic pods. This eliminates unnecessary connections to the database – Keeping these connections open can result locks or timeouts as the script runs.

As an example, the following function can perform this task:

```
function turbo_stop_all_pods {
# First we need to stop t8c operator
  echo "Stopping t8c-operator..."
  turbo_stop_all_pods=$(kubectl scale deployment t8c-operator --replicas=0 -n turbonomi
c)
  sleep 3
  # Now we stop all other pods
  turbo_stop_all_pods=$(kubectl get deploy -n turbonomic --no-headers=true | cut -d ' '
 -f1 | xargs -I % kubectl scale --replicas=0 deployment/% -n turbonomic)
  until [[ $(kubectl get pods -n turbonomic --no-headers=true | grep -cvE 'glusterfs|g
rafana|prometheus') -eq 0  ]] ; do
    echo -e "turbo_stop_all_pods: Waiting on Turbonomic pods to terminate, so far: \n
$(kubectl get pods -n turbonomic --no-headers=true | grep -v 'glusterfs|grafana|prometh
eus')"
    sleep 3
  done
  echo "All Turbonomic pods are Terminated - exiting"
```

Executing the Data Migration

To migrate the data, you will run the script and then restart the Turbonomic platform:

1. Install the data migration script.

   Download the script from the following location:

   https://github.com/turbonomic/t8c-install/blob/master/bin/convertMaxUtilizationToSettingOverride.py

2. Run the data migration script.

   You can run the script from any machine that has access to your installed Turbonomic platform. For example, if your local machine can execute `kubectl` on the Turbonomic platform, then you can run the script from that local machine. You can also install the script on the VM that hosts the platform, and run it from there.

   To run the script, execute the command:

   ```
   convertMaxUtilizationToSettingOverride.py
   ```

   The script will require the following arguments:

   - user name
   - password
   - host address – The VM that hosts the Turbonomic platform, or that hosts the database (for a remote database installation)
   - port
   - database name

   As an example, the command should be similar to:

```
python convertMaxUtilizationToSettingOverride.py root vmturbo localhost 3306 vmtdb
```

As the script runs, it posts entries to the log file, `convert_utilization_settings_{currentDate}.log`. The entries list how many rows per entity type the script has migrated. The script migrates all the data for an entity type in a single transaction. If the script exits without errors, it has succeeded.

3. Restart the Turbonomic Platform

   To restart the platform, scale the operator back up to one replica. For example:

   ```
   kubectl scale deployment -n turbonomic t8c-operator --replicas=1
   ```

Appendix A: Turbonomic Components

The following listings show the components that Turbonomic creates as part of the installation process.

Core Components For Turbonomic

- `rsyslog`
- `nginx`
- `consul`
- `auth`
- `clustermgr`
- `api`
- `market`
- `action-orchestrator`
- `topology-processor`
- `arangodb`
- `repository`
- `group`
- `history`
- `plan-orchestrator`

Mediation Components For Turbonomic

- `mediation-actionscript`
- `mediation-appdynamics`
- `mediation-hpe3par`
- `mediation-hyperv`
- `mediation-netapp`
- `mediation-oneview`
- `mediation-pure`
- `mediation-ucs`
- `mediation-vcenter`

- `mediation-vcenterbrowsing`
- `mediation-vmax`
- `mediation-vmm`

Appendix B: What Are the Typical Settings for an IdP?

Before you begin configuring Single Sign-On (SSO), you need to make sure the IdP is set up for SSO.

Here are typical settings for a public Okta IdP which may be useful when you set up your IdP.

SAML Settings: GENERAL	
Setting	**Example**
Single Sign On URL	`https://10.10.10.123/vmturbo/saml/SSO`
Recipient URL	`https://10.10.10.123/vmturbo/saml/SSO`
Destination URL	`https://10.10.10.123/vmturbo/saml/SSO`
Audience Restriction	`urn:test:turbo:markharm`
Default Relay State	
Name ID Format	`Unspecified`
Application username	The username for the account that is managed by Okta
Response	`Signed`
Assertion Signature	`Signed`
Signature Algorithm	`RSA_SHA256`
Digital Algorithm	`SHA256`
Assertion Encryption	`Unencrypted`
SAML Single Logout	`Enabled`
Single Logout URL	`https://10.10.10.123/vmturbo/rest/logout`
SP Issuer	`turbo`

SAML Settings: GENERAL	
Setting	**Example**
Signature Certificate	`Example.cer (CN=apollo)`
authnContextClassRef	`PasswordProtectedTransport`
Honor Force Authentication	`Yes`
SAML Issuer ID	`http://www.okta.com/$(org.externalKey)`

SAML Settings: GROUP ATTRIBUTE STATEMENTS		
Name	**Name Format**	**Filter**
`group`	`Unspecified`	`Matches regex:.*admin.*.`