

Getting Started with Trento

SUSE® Linux Enterprise Server for SAP Applications

Trento is an open cloud native Web console that aims to help SAP Basis consultants and administrators to check the configuration, monitor and manage the entire OS stack of their SAP environments, including HA features.

Publication Date: 05 Feb 2025

Contents

- 1 What is Trento? 3
- 2 Product lifecycle and update strategy 5
- 3 Requirements 5
- 4 Installing Trento Server 7
- 5 Installing Trento Agents 30
- 6 User management 32
- 7 Single Sign-On integration 34
- 8 Activity Log 47
- 9 Performing configuration checks 48
- 10 Using Trento Web console 51
- 11 Housekeeping 63
- 12 Managing tags 64
- 13 Integration with SUSE Manager 66

14	Rotating API keys	71
15	Updating Trento Server	71
16	Updating a Trento Agent	73
17	Updating Trento Checks	74
18	Uninstalling Trento Server	75
19	Uninstalling a Trento Agent	75
20	Reporting a Problem	75
21	Problem Analysis	76
22	Compatibility matrix between Trento Server and Trento Agents	80
23	Highlights of Trento versions	80
24	More information	84

1 What is Trento?

Trento is the official version of the Trento community project. It is a comprehensive monitoring solution consisting of two main components: the Trento Server and the Trento Agent. Trento provides the following functionality and features:

- A user-friendly reactive Web interface for SAP Basis administrators.
- Automated discovery of Pacemaker clusters using SAPHanaSR classic or angi as well as different fencing mechanisms, including diskless SBD.
- Automated discovery of SAP systems running on ABAP or JAVA stacks and HANA databases.
- Awareness of maintenance situations in a Pacemaker cluster at cluster, node, or resource level.
- Configuration validation for SAP HANA Scale-Up Performance-optimized, SAP HANA Scale-out and ASCS/ERS clusters deployed on Azure, AWS, GCP or on-premises bare metal platforms, including KVM and Nutanix.
- Useful information that offers insights about the execution of configuration checks.
- Delivery of configuration checks decoupled from core functionality.
- Email alerting for critical events in the monitored landscape.
- Integration of saptune into the console and specific configuration checks at host and cluster levels.
- Information about relevant patches and upgradable packages for registered hosts via integration with SUSE Manager.
- Monitoring of CPU and memory usage at the host level through basic integration with Prometheus.
- API-based architecture to facilitate integration with other monitoring tools.
- Rotating API key to protect communication from the Trento Agent to the Trento Server.
- Housekeeping capabilities.

The **Trento Server** is an independent, distributed system designed to run on a Kubernetes cluster or as a regular systemd stack. The Trento Server provides a Web front-end for user interaction. The Trento Server consists of the following components:

- The web component that acts as a control plane responsible for internal and external communications as well as rendering the UI.
- The checks engine named wanda that orchestrates the execution of configuration checks.
- A PostgreSQL database for data persistence.
- The RabbitMQ message broker for communicating between the checks engine and the agents.
- A Prometheus instance that retrieves the metrics collected by the Prometheus node exporter in the registered hosts.

The **Trento Agent** is a single background process (trento_agent) running on each monitored host of the SAP infrastructure.

See *Figure 1, "Architectural overview"* for additional details.

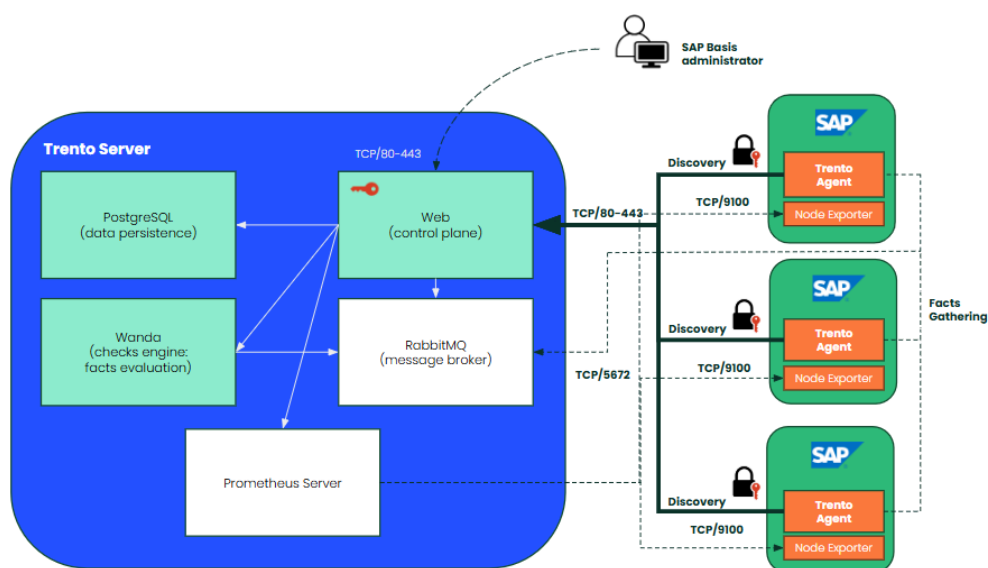


FIGURE 1: ARCHITECTURAL OVERVIEW

2 Product lifecycle and update strategy

Trento is available at no extra cost for SUSE Linux Enterprise Server for SAP Applications 15 subscribers. Particularly, Trento's two main components have the following product lifecycles:

Trento Agent

Delivery mechanism: RPM package for SUSE Linux Enterprise Server for SAP Applications 15 SP3 and newer.

Supported runtime: Supported in SUSE Linux Enterprise Server for SAP Applications 15 SP3 and newer on x86_64 and ppc64le architectures.

Trento Server

Delivery mechanisms: A set of container images from the SUSE public registry together with a Helm chart that facilitates their installation or a set of RPM packages for SUSE Linux Enterprise Server for SAP Applications 15 SP3 and newer.

Kubernetes deployment: The Trento Server runs on any current Cloud Native Computing Foundation (CNCF)-certified Kubernetes distribution based on a x86_64 architecture. Depending on your scenario and needs, SUSE supports several usage scenarios:

- If you already use a CNCF-certified Kubernetes cluster, you can run the Trento Server in it.
- If you don't have a Kubernetes cluster, and need enterprise support, SUSE recommends SUSE Rancher Kubernetes Engine (RKE) version 1 or 2.
- If you do not have a Kubernetes enterprise solution but you want to try Trento, SUSE Rancher's K3s provides you with an easy way to get started. But keep in mind that K3s default installation process deploys a single node Kubernetes cluster, which is not a recommended setup for a stable Trento production instance.

systemd and containerized deployments: Supported in SUSE Linux Enterprise Server for SAP Applications 15 SP3 and newer.

3 Requirements

This section describes requirements for the Trento Server and its Trento Agents.

3.1 Trento Server requirements

Running all the Trento Server components requires a minimum of 4 GB of RAM, two CPU cores and 64 GB of storage. When using K3s, such storage should be provided under `/var/lib/rancher/k3s`.

Trento is based on an event-driven technology. Registered events are stored in a PostgreSQL database with a default retention period of 10 days. For each host registered with Trento, you need to allocate at least 1.5GB of space in the PostgreSQL database.

Trento Server supports different deployment scenarios: Kubernetes, systemd, and containers. A Kubernetes-based deployment of Trento Server is cloud-native and OS-agnostic. It can be performed on the following services:

- RKE1 (Rancher Kubernetes Engine version 1)
- RKE2
- a Kubernetes service in a cloud provider
- any other CNCF-certified Kubernetes running on x86_64 architecture

A production-ready Kubernetes-based deployment of Trento Server requires Kubernetes knowledge. The Helm chart is intended to be used by customers without in-house Kubernetes expertise, or as a way to try Trento with a minimum of effort. However, Helm chart delivers a basic deployment of the Trento Server with all the components running on a single node of the cluster.

3.2 Trento Agent requirements

The resource footprint of the Trento Agent is designed to not impact the performance of the host it runs on.

The Trento Agent component needs to interact with several low-level system components that are part of the SUSE Linux Enterprise Server for SAP Applications distribution.

The hosts must have unique machine identifiers (ids) in order to be registered in Trento. This means that if a host in your environment is built as a clone of another one, make sure to change the machine's identifier as part of the cloning process before starting the Trento Agent on it.

Similarly, the clusters must have unique authkeys on the clusters in order to be registered in Trento.

3.3 Network requirements

- The Web component of the Trento Server must be reachable from any Trento Agent host via HTTP (port TCP/80) or via HTTPS (port TCP/443) if SSL is enabled.
- The checks engine component of the Trento Server must be reachable from any Trento Agent host via Advanced Message Queuing Protocol or AMQP (port TCP/5672).
- The Prometheus component of the Trento Server must be able to reach the Node Exporter in the Trento Agent hosts (port TCP/9100).
- The SAP Basis administrator must access to the Web component of the Trento Server via HTTP (port TCP/80) or via HTTPS (port TCP/443) if SSL is enabled.

3.4 Installation prerequisites

- **Trento Server.** For a Kubernetes-based deployment, you must have access to SUSE public registry for the deployment of Trento Server containers. For a systemd deployment, you must have a registered SUSE Linux Enterprise Server for SAP Applications 15 (SP3 or higher) distribution. The same applies to a containerized deployment.
- **Trento Agents.** A registered SUSE Linux Enterprise Server for SAP Applications 15 (SP3 or higher) distribution.

4 Installing Trento Server

4.1 Kubernetes deployment

The subsection uses the following placeholders:

- TRENTO_SERVER_HOSTNAME: the host name used by the end user to access the console.
- ADMIN_PASSWORD: the password of the default user created during the installation process. It must have at least 8 characters.

4.1.1 Installing Trento Server on an existing Kubernetes cluster

Trento Server consists of a several components delivered as container images and intended for deployment on a Kubernetes cluster. A manual production-ready deployment of these components requires Kubernetes knowledge. Customers without in-house Kubernetes expertise and who want to try Trento with a minimum of effort, can use the Trento Helm chart. This approach automates the deployment of all the required components on a single Kubernetes cluster node. You can use the Trento Helm chart to install Trento Server on a existing Kubernetes cluster as follows:

1. Install Helm:

```
curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash
```

2. Connect Helm to an existing Kubernetes cluster.
3. Use Helm to install Trento Server with the Trento Helm chart:

```
helm upgrade \
  --install trento-server oci://registry.suse.com/trento/trento-server \
  --set trento-web.trentoWeb0origin=TRENTO_SERVER_HOSTNAME \
  --set trento-web.adminUser.password=ADMIN_PASSWORD
```

When using a Helm version lower than 3.8.0, an experimental flag must be set as follows:

```
HELM_EXPERIMENTAL_OCI=1 helm upgrade \
  --install trento-server oci://registry.suse.com/trento/trento-server \
  --set trento-web.trentoWeb0origin=TRENTO_SERVER_HOSTNAME \
  --set trento-web.adminUser.password=ADMIN_PASSWORD
```

4. To verify that the Trento Server installation was successful, open the URL of the Trento Web console (http://TRENTO_SERVER_HOSTNAME) from a workstation on the SAP administrator's LAN.

4.1.2 Installing Trento Server on K3s

If you do not have a Kubernetes cluster, or have one but do not want to use it for Trento, SUSE Rancher's K3s provides an alternative. To deploy Trento Server on K3s, you need is a small server or VM (see [Section 3.1, "Trento Server requirements"](#) for minimum requirements) and follow steps in [Procedure 1, "Manually installing Trento on a Trento Server host"](#).

! Important: Deploying Trento on K3s

The following procedure deploys Trento Server on a single-node K3s cluster. Note that this setup is not recommended for production use.

PROCEDURE 1: MANUALLY INSTALLING TRENTO ON A TRENTO SERVER HOST

1. Log in to the Trento Server host.

2. Install K3s:

- Installing as user root

```
# curl -sL https://get.k3s.io | INSTALL_K3S_SKIP_SELINUX_RPM=true sh
```

- Installing as non-root user:

```
> curl -sL https://get.k3s.io | INSTALL_K3S_SKIP_SELINUX_RPM=true sh -s - --  
write-kubeconfig-mode 644
```

3. Install Helm as root:

```
# curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash
```

4. Set the KUBECONFIG environment variable for the same user that installed K3s:

```
export KUBECONFIG=/etc/rancher/k3s/k3s.yaml
```

5. With the same user that installed K3s, install Trento Server using the Helm chart:

```
helm upgrade \  
  --install trento-server oci://registry.suse.com/trento/trento-server \  
  --set trento-web.trentoWeb0origin=TRENTO_SERVER_HOSTNAME \  
  --set trento-web.adminUser.password=ADMIN_PASSWORD
```

When using a Helm version lower than 3.8.0, an experimental flag must be set as follows:

```
HELM_EXPERIMENTAL_OCI=1 helm upgrade \  
  --install trento-server oci://registry.suse.com/trento/trento-server \  
  --set trento-web.trentoWeb0origin=TRENTO_SERVER_HOSTNAME \  
  --set trento-web.adminUser.password=ADMIN_PASSWORD
```

6. Monitor the creation and start-up of the Trento Server pods, and wait until they are ready and running:

```
watch kubectl get pods
```

All pods must be in the ready and running state.

7. Log out of the Trento Server host.
8. To verify that the Trento Server installation was successful, open the URL of the Trento Web console (http://TRENTO_SERVER_HOSTNAME) from a workstation on the SAP administrator's LAN.

4.1.3 Deploying Trento Server on selected nodes

If you use a multi-node Kubernetes cluster, it is possible to deploy Trento Server images on selected nodes by specifying the field `nodeSelector` in the helm upgrade command as follows:

```
HELM_EXPERIMENTAL_OCI=1 helm upgrade \
--install trento-server oci://registry.suse.com/trento/trento-server \
--set trento-web.trentoWebOrigin=TRENTO_SERVER_HOSTNAME \
--set trento-web.adminUser.password=ADMIN_PASSWORD \
--set prometheus.server.nodeSelector.LABEL=VALUE \
--set postgresql.primary.nodeSelector.LABEL=VALUE \
--set trento-web.nodeSelector.LABEL=VALUE \
--set trento-runner.nodeSelector.LABEL=VALUE
```

4.1.4 Configuring event pruning

The event pruning feature allows administrators to manage how long registered events are stored in the database and how often the expired events are removed.

The following configuration options are available:

pruneEventsOlderThan

The number of days registered events are stored in the database. The default value is **10**. *Keep in mind that `pruneEventsOlderThan` can be set to **0**. However, this deletes all events whenever the cron job runs, making it impossible to analyze and troubleshoot issues with the application*

pruneEventsCronjobSchedule

The frequency of the cron job that deletes expired events. The default value is "0 0 * * *", which runs daily at midnight.

To modify the default values, execute the following Helm command:

```
helm ... \
  --set trento-web.pruneEventsOlderThan=<<EXPIRATION_IN_DAYS>> \
  --set trento-web.pruneEventsCronjobSchedule="<<NEW_SCHEDULE>>"
```

Replace the placeholders with the desired values:

EXPIRATION_IN_DAYS

Number of days to retain events in the database before pruning.

NEW_SCHEDULE

The cron rule specifying how frequently the pruning job is performed.

Example command to retain events for 30 days and schedule pruning daily at 3 AM:

```
helm upgrade \
  --install trento-server oci://registry.suse.com/trento/trento-server \
  --set trento-web.trentoWebOrigin=TRENTO_SERVER_HOSTNAME \
  --set trento-web.adminUser.password=ADMIN_PASSWORD \
  --set trento-web.pruneEventsOlderThan=30 \
  --set trento-web.pruneEventsCronjobSchedule="0 3 * * *"
```

4.1.5 Enabling email alerts

Email alerting feature notifies the SAP Basis administrator about important changes in the SAP Landscape being monitored by Trento.

The reported events include the following:

- Host heartbeat failed
- Cluster health detected critical
- Database health detected critical
- SAP System health detected critical

This feature is disabled by default. It can be enabled at installation time or anytime at a later stage. In both cases, the procedure is the same and uses the following placeholders:

SMTP_SERVER

The SMTP server designated to send email alerts

SMTP_PORT

Port on the SMTP server

SMTP_USER

User name to access SMTP server

SMTP_PASSWORD

Password to access SMTP server

ALERTING_SENDER

Sender email for alert notifications

ALERTING_RECIPIENT

Recipient email for alert notifications.

The command to enable email alerts is as follows:

```
HELM_EXPERIMENTAL_OCI=1 helm upgrade \
--install trento-server oci://registry.suse.com/trento/trento-server \
--set trento-web.trentoWebOrigin=TRENTO_SERVER_HOSTNAME \
--set trento-web.adminUser.password=ADMIN_PASSWORD \
--set trento-web.alerting.enabled=true \
--set trento-web.alerting.smtpServer=SMTP_SERVER \
--set trento-web.alerting.smtpPort=SMTP_PORT \
--set trento-web.alerting.smtpUser=SMTP_USER \
--set trento-web.alerting.smtpPassword=SMTP_PASSWORD \
--set trento-web.alerting.sender=ALERTING_SENDER \
--set trento-web.alerting.recipient=ALERTING_RECIPIENT
```

4.1.6 Enabling SSL

Ingress may be used to provide SSL termination for the Web component of Trento Server. This would allow to encrypt the communication from the agent to the server, which is already secured by the corresponding API key. It would also allow HTTPS access to the Web console with trusted certificates.

Configuration must be done in the `tls` section of the `values.yaml` file of the chart of the Trento Server Web component.

For details on the required Ingress setup and configuration, refer to: <https://kubernetes.io/docs/concepts/services-networking/ingress/>. Particularly, refer to section <https://kubernetes.io/docs/concepts/services-networking/ingress/#tls> for details on the secret format in the YAML configuration file.

Additional steps are required on the Agent side.

4.2 systemd deployment

A systemd installation of Trento Server, based on RPM packages, can be done manually. The latest available version of SUSE Linux Enterprise Server for SAP Applications is used as the base operating system, which is SLE 15 SP5 (<https://www.suse.com/download/sles/>) at the time of writing. For installations on Service Packs other than SP5, update the repository address as indicated in the respective notes provided throughout this guide.

Supported Service Packs:


- SP3
- SP4
- SP5

4.2.1 List of dependencies

- PostgreSQL (<https://www.postgresql.org/>)
- RabbitMQ (<https://rabbitmq.com/>)
- NGINX (<https://nginx.org/en/>)
- Prometheus (<https://prometheus.io/>) (optional)

4.2.2 Install Trento dependencies

4.2.2.1 Install Prometheus (Optional)

Prometheus (<https://prometheus.io/>)  is not required to run Trento, but it is recommended, as it allows Trento to display charts for each host with useful information about the CPU load, memory, and other important metrics.




Note


If you choose not to install Prometheus, or you choose to use an existing installation, make sure that `CHARTS_ENABLED` is set to `false` in the Trento web RPM configuration file stored at `/etc/trento/trento-web`, or when it is provided to the Trento web container.

4.2.2.1.1 Option 1: Use existing installation

Minimal required Prometheus version is **2.28.0**.

If you have a [existing Prometheus server \(https://prometheus.io/docs/prometheus/latest/installation/\)](https://prometheus.io/docs/prometheus/latest/installation/) , ensure to set the `PROMETHEUS_URL` environment variable to your Prometheus server's URL as part of the Docker command when creating the `trento-web` container or configuring the RPM packages. Use [Section 4.2.2.1.2, "Option 2: Install Prometheus using the **unsupported** PackageHub repository"](#) as a reference to adjust the Prometheus configuration.

4.2.2.1.2 Option 2: Install Prometheus using the **unsupported** PackageHub repository

PackageHub (<https://packagehub.suse.com/>)  packages are tested by SUSE, but they do not come with the same level of support as the core SLES packages. Users should assess the suitability of these packages based on their own risk tolerance and support needs.

1. Enable PackageHub repository:

```
SUSEConnect --product PackageHub/15.5/x86_64
```



Note

SLE15 SP3 requires a provided Prometheus server. The version available through `SUSEConnect --product PackageHub/15.3/x86_64` is outdated and is not compatible with Trento's Prometheus configuration. Refer to [Section 4.2.2.1.1, "Option 1: Use existing installation"](#) for SLE 15 SP3.



Note

For SLE15 SP4 change the repository to `SUSEConnect --product PackageHub/15.4/x86_64`

2. Add the Prometheus user/group:

```
groupadd --system prometheus
useradd -s /sbin/nologin --system -g prometheus prometheus
```

3. Install Prometheus using Zypper:

```
zypper in golang-github-prometheus-prometheus
```



Note

In case the missing dependency cannot be satisfied, we have already added the Prometheus user/group. This makes it safe to proceed with the installation by choosing Solution 2: `break golang-github-prometheus-prometheus`

4. Change Prometheus configuration by replacing the configuration at `/etc/prometheus/prometheus.yml` with:

```
global:
  scrape_interval: 30s
  evaluation_interval: 10s

scrape_configs:
  - job_name: "http_sd_hosts"
    honor_timestamps: true
    scrape_interval: 30s
    scrape_timeout: 30s
    scheme: http
```

```
follow_redirects: true
http_sd_configs:
  - follow_redirects: true
    refresh_interval: 1m
    url: http://localhost:4000/api/prometheus/targets
```



Note

localhost:4000 in **url:** <http://localhost:4000/api/prometheus/targets> refers to the location where Trento web service is running.

5. Enable and start the Prometheus service:

```
systemctl enable --now prometheus
```

6. If firewalld is running, allow Prometheus to be accessible and add an exception to firewall:

```
firewall-cmd --zone=public --add-port=9090/tcp --permanent
firewall-cmd --reload
```

4.2.2.2 Install PostgreSQL

The current instructions are tested with the following PostgreSQL version:

- 13.9 for SP3
- 14.10 for SP4
- 15.5 for SP5

Using a different version of PostgreSQL may require different steps or configurations, especially when changing the major number. For more details, refer to the official [PostgreSQL documentation](https://www.postgresql.org/docs/) (<https://www.postgresql.org/docs/>) [↗](#).

1. Install PostgreSQL server:

```
zypper in postgresql-server
```

2. Enable and start PostgreSQL server:

```
systemctl enable --now postgresql
```


4.2.2.2.1 Configure PostgreSQL

1. Start `psql` with the `postgres` user to open a connection to the database:

```
su - postgres
psql
```

2. Initialize the databases in the `psql` console:

```
CREATE DATABASE wanda;
CREATE DATABASE trento;
CREATE DATABASE trento_event_store;
```

3. Create the users:

```
CREATE USER wanda_user WITH PASSWORD 'wanda_password';
CREATE USER trento_user WITH PASSWORD 'web_password';
```

4. Grant required privileges to the users and close the connection:

```
\c wanda
GRANT ALL ON SCHEMA public TO wanda_user;
\c trento
GRANT ALL ON SCHEMA public TO trento_user;
\c trento_event_store;
GRANT ALL ON SCHEMA public TO trento_user;
\q
```

You can exit from the `psql` console and `postgres` user.

5. Allow the PostgreSQL database to receive connections to the respective databases and users. To do this, add the following to `/var/lib/pgsql/data/pg_hba.conf`:

```
host    wanda                                wanda_user    0.0.0.0/0      md5
host    trento,trento_event_store          trento_user    0.0.0.0/0      md5
```



Note

The `pg_hba.conf` file works sequentially. This means that the rules on the top have preference over the ones below. The example above shows a permissive address range. So for this to work, the entires must be written at the top of the `host` entries. For further information, refer to the `pg_hba.conf` (<https://www.postgresql.org/docs/current/auth-pg-hba-conf.html>) [↗](#) documentation.

6. Allow PostgreSQL to bind on all network interfaces in `/var/lib/pgsql/data/postgresql.conf` by changing the following line:

```
listen_addresses = '*'
```

7. Restart PostgreSQL to apply the changes:

```
systemctl restart postgresql
```

4.2.2.3 Install RabbitMQ

1. Install RabbitMQ server:

```
zypper install rabbitmq-server
```

2. Allow connections from external hosts by modifying `/etc/rabbitmq/rabbitmq.conf`, so the Trento-agent can reach RabbitMQ:

```
listeners.tcp.default = 5672
```

3. If firewalld is running, add a rule to firewalld:

```
firewall-cmd --zone=public --add-port=5672/tcp --permanent;  
firewall-cmd --reload
```

4. Enable the RabbitMQ service:

```
systemctl enable --now rabbitmq-server
```

4.2.2.3.1 Configure RabbitMQ

To configure RabbitMQ for a production system, follow the official suggestions in the [RabbitMQ guide \(https://www.rabbitmq.com/production-checklist.html\)](https://www.rabbitmq.com/production-checklist.html) ↗.

1. Create a new RabbitMQ user:

```
rabbitmqctl add_user trento_user trento_user_password
```

2. Create a virtual host:

```
rabbitmqctl add_vhost vhost
```

3. Set permissions for the user on the virtual host:

```
rabbitmqctl set_permissions -p vhost trento_user ".*" ".*" ".*"
```

4.2.3 Install Trento using RPM packages

The `trento-web` and `trento-wanda` packages come in the supported SLES4SAP distributions by default.

Install Trento web and wanda:

```
zypper install trento-web trento-wanda
```

4.2.3.1 Create the configuration files

Both services depend on respective configuration files. They must be placed in `/etc/trento/trento-web` and `/etc/trento/trento-wanda` respectively, and examples of how to modify them are available in `/etc/trento/trento-web.example` and `/etc/trento/trento-wanda.example`.

Important: The content of `SECRET_KEY_BASE` and `ACCESS_TOKEN_ENC_SECRET` in both `trento-web` and `trento-wanda` must be the same.



Note

You can create the content of the secret variables like `SECRET_KEY_BASE`, `ACCESS_TOKEN_ENC_SECRET` and `REFRESH_TOKEN_ENC_SECRET` with `openssl` running `openssl rand -out /dev/stdout 48 | base64`



Note

Depending on how you intend to connect to the console, a working hostname, FQDN, or an IP is required in `TRENTO_WEB_ORIGIN` for HTTPS. Otherwise websockets fail to connect, causing no real-time updates in the UI.

4.2.3.2 trento-web configuration

```
# /etc/trento/trento-web
```

```
AMQP_URL=amqp://trento_user:trento_user_password@localhost:5672/vhost
DATABASE_URL=ecto://trento_user:web_password@localhost/trento
EVENTSTORE_URL=ecto://trento_user:web_password@localhost/trento_event_store
ENABLE_ALERTING=false
CHARTS_ENABLED=true
PROMETHEUS_URL=http://localhost:9090
ADMIN_USER=admin
ADMIN_PASSWORD=test1234
ENABLE_API_KEY=true
PORT=4000
TRENTO_WEB_ORIGIN=trento.example.com
SECRET_KEY_BASE=some-secret
ACCESS_TOKEN_ENC_SECRET=some-secret
REFRESH_TOKEN_ENC_SECRET=some-secret
```



Note

Note: Add `CHARTS_ENABLED=false` in Trento web configuration file if Prometheus is not installed or you do not want to use Trento's charts functionality.

The [alerting system to receive email notifications \(https://github.com/trento-project/web/blob/main/guides/alerting/alerting.md\)](https://github.com/trento-project/web/blob/main/guides/alerting/alerting.md) can be enabled by setting `ENABLE_ALERTING` to `true` and adding the following entries:

```
# /etc/trento/trento-web
ENABLE_ALERTING=true
ALERT_SENDER=<<SENDER_EMAIL_ADDRESS>>
ALERT_RECIPIENT=<<RECIPIENT_EMAIL_ADDRESS>>
SMTP_SERVER=<<SMTP_SERVER_ADDRESS>>
SMTP_PORT=<<SMTP_PORT>>
SMTP_USER=<<SMTP_USER>>
SMTP_PASSWORD=<<SMTP_PASSWORD>>
```

4.2.3.3 trento-wanda configuration

```
# /etc/trento/trento-wanda
CORS_ORIGIN=http://localhost
AMQP_URL=amqp://trento_user:trento_user_password@localhost:5672/vhost
DATABASE_URL=ecto://wanda_user:wanda_password@localhost/wanda
PORT=4001
SECRET_KEY_BASE=some-secret
ACCESS_TOKEN_ENC_SECRET=some-secret
```

4.2.3.4 Start the services

Enable and start the services:

```
systemctl enable --now trento-web trento-wanda
```

4.2.3.5 Monitor the services

Use `journalctl` to check if the services are up and running correctly. For example:

```
journalctl -fu trento-web
```

4.2.4 Check the health status of trento web and wanda

You can check if Trento web and wanda services function correctly by accessing the `healthz` and `readyz` API.

1. Check Trento web health status using `curl`:

```
curl http://localhost:4000/api/readyz
```

```
curl http://localhost:4000/api/healthz
```

2. Check Trento wanda health status using `curl`:

```
curl http://localhost:4001/api/readyz
```

```
curl http://localhost:4001/api/healthz
```

If Trento web and wanda are ready, and the database connection is set up correctly, the output should be as follows:

```
{"ready":true>{"database":"pass"}
```

4.2.5 Install and configure NGINX

1. Install NGINX package:

```
zypper install nginx
```

2. If firewalld is running, add firewalld rules for HTTP and HTTPS:

```
firewall-cmd --zone=public --add-service=https --permanent
firewall-cmd --zone=public --add-service=http --permanent
firewall-cmd --reload
```

3. Start and enable NGINX:

```
systemctl enable --now nginx
```

4. Create a configuration file for Trento:

```
vim /etc/nginx/conf.d/trento.conf
```

5. Add the following configuration to /etc/nginx/conf.d/trento.conf:

```
server {
    # Redirect HTTP to HTTPS
    listen 80;
    server_name trento.example.com;
    return 301 https://$host$request_uri;
}

server {
    # SSL configuration
    listen 443 ssl;
    server_name trento.example.com;

    ssl_certificate /etc/nginx/ssl/certs/trento.crt;
    ssl_certificate_key /etc/ssl/private/trento.key;

    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers 'ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-
    ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-
    RSA-AES256-GCM-SHA384';
    ssl_prefer_server_ciphers on;
    ssl_session_cache shared:SSL:10m;

    # Wanda rule
    location ~ ^/(api/checks|api/v1/checks|api/v2/checks|api/v3/checks)/ {
        allow all;
    }

    # Proxy Headers
    proxy_http_version 1.1;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header Host $http_host;
    proxy_set_header X-Cluster-Client-Ip $remote_addr;
```

```

        # Important Websocket Bits!
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";

        proxy_pass http://localhost:4001;
    }

    # Web rule
    location / {
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;

        # The Important Websocket Bits!
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";

        proxy_pass http://localhost:4000;
    }
}

```

6. Check the NGINX configuration:

```
nginx -t
```

If the configuration is correct, the output should be as follows:

```

nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful

```

If there are issues with the configuration, the output indicates what needs to be adjusted.

4.2.6 Prepare SSL certificate for NGINX

Create or provide a certificate for [NGINX \(https://nginx.org/en/\)](https://nginx.org/en/)  to enable SSL for Trento.

4.2.6.1 Create a self-signed certificate

1. Generate a self-signed certificate:



Note

Adjust `subjectAltName = DNS:trento.example.com` by replacing `trento.example.com` with your domain and change the value `5` to the number of days for which you need the certificate to be valid. For example, `-days 365` for one year.

```
openssl req -newkey rsa:2048 --nodes -keyout trento.key -x509 -days 5 -out  
trento.crt -addext "subjectAltName = DNS:trento.example.com"
```

2. Copy the generated `trento.key` to a location accessible by NGINX:

```
cp trento.key /etc/ssl/private/trento.key
```

3. Create a directory for the generated `trento.crt` file. The directory must be accessible by NGINX:

```
mkdir -p /etc/nginx/ssl/certs/
```

4. Copy the generated `trento.crt` file to the created directory:

```
cp trento.crt /etc/nginx/ssl/certs/trento.crt
```

5. Reload NGINX to apply changes:

```
systemctl reload nginx
```

4.2.6.2 Create a signed certificate with Let's Encrypt using PackageHub repository



Note

Change repository if you use a Service Pack other than SP5. For example: `SUSEConnect --product PackageHub/15.3/x86_64` for SLE15 SP3, `SUSEConnect --product PackageHub/15.4/x86_64` for SLE15 SP4. Use packages in PackageHub at your own risk.

1. Add PackageHub, if it is not already added:

```
SUSEConnect --product PackageHub/15.5/x86_64  
zypper refresh
```

2. Install Certbot and its NGINX plugin:

```
zypper install certbot python3-certbot-nginx
```

3. Obtain a certificate and configure NGINX with Certbot:



Note

Replace `example.com` with your domain. For more information, refer to [Certbot instructions for NGINX \(https://certbot.eff.org/instructions?ws=nginx&os=leap\)](https://certbot.eff.org/instructions?ws=nginx&os=leap) ↗

```
certbot --nginx -d example.com -d www.example.com
```



Note

Certbot certificates are valid for 90 days. Refer to the above link for details on how to renew certificates.

4.2.7 Accessing the trento-web UI

Pin the browser to `https://trento.example.com`. You should be able to login using the credentials specified in the `ADMIN_USER` and `ADMIN_PASSWORD` environment variables.

4.3 Containerized deployment

A containerized deployment of Trento Server is identical to a systemd deployment. However, the web and check engine components are deployed as Docker containers.

Follow the steps in [Section 4.2, “systemd deployment”](#), but skip the **Install Trento using RPM packages** step and follow the procedures in [Section 4.3.1, “Install Trento using Docker”](#).

4.3.1 Install Trento using Docker

4.3.1.1 Install Docker container runtime

1. Enable the containers module:

```
SUSEConnect --product sle-module-containers/15.5/x86_64
```



Note

To use a different Service Pack than SP5, you have to choose the appropriate repository. For example, **SUSEConnect --product sle-module-containers/15.3/x86_64** for SLE15 SP3, **SUSEConnect --product sle-module-containers/15.4/x86_64** for SLE15 SP4.

2. Install Docker:

```
zypper install docker
```

3. Enable and start Docker:

```
systemctl enable --now docker
```


4.3.1.2 Create a dedicated Docker network for Trento

1. Create the Trento Docker network:

```
docker network create trento-net
```



Note

When creating the `trento-net` network, Docker assigns a default subnet to it: `172.17.0.0/16`. Ensure that this subnet is allowed by the rules specified in your PostgreSQL configuration. For more information, refer to the upstream [pg_hba.conf](https://www.postgresql.org/docs/current/auth-pg-hba-conf.html) (<https://www.postgresql.org/docs/current/auth-pg-hba-conf.html>)  documentation.

2. Verify the subnet of `trento-net`:

```
docker network inspect trento-net --format '{{range .IPAM.Config}}{{.Subnet}}
{{end}}'
```

Expected output is as follows:

```
172.17.0.0/16
```

4.3.1.3 Install Trento on Docker

1. Create secret environment variables:



Note

Consider using an environment variable file (see [official Docker documentation](https://docs.docker.com/engine/reference/commandline/run/#env) (<https://docs.docker.com/engine/reference/commandline/run/#env>)). Adjust the docker command below for use with the env file. In any case, make sure you keep a copy of the generated keys in a safe location, in case you need to reuse them in the future.

```
WANDA_SECRET_KEY_BASE=$(openssl rand -out /dev/stdout 48 | base64)
TRENTO_SECRET_KEY_BASE=$(openssl rand -out /dev/stdout 48 | base64)
ACCESS_TOKEN_ENC_SECRET=$(openssl rand -out /dev/stdout 48 | base64)
REFRESH_TOKEN_ENC_SECRET=$(openssl rand -out /dev/stdout 48 | base64)
```

2. Install the checks on the system in a shared volume:

```
docker volume create trento-checks \
&& docker run \
-v trento-checks:/usr/share/trento/checks \
registry.suse.com/trento/trento-checks:latest
```

3. Deploy trento-wanda:

```
docker run -d --name wanda \
-p 4001:4000 \
--network trento-net \
--add-host "host.docker.internal:host-gateway" \
-v trento-checks:/usr/share/trento/checks:ro \
-e CORS_ORIGIN=localhost \
```

```
-e SECRET_KEY_BASE=$WANDA_SECRET_KEY_BASE \
-e ACCESS_TOKEN_ENC_SECRET=$ACCESS_TOKEN_ENC_SECRET \
-e AMQP_URL=amqp://trento_user:trento_user_password@host.docker.internal/vhost \
-e DATABASE_URL=ecto://wanda_user:wanda_password@host.docker.internal/wanda \
--restart always \
--entrypoint /bin/sh \
registry.suse.com/trento/trento-wanda:latest \
-c "/app/bin/wanda eval 'Wanda.Release.init()' && /app/bin/wanda start"
```

4. Deploy trento-web.

Make sure to change the `ADMIN_USER` and `ADMIN_PASSWORD`, these are the credentials that are required to login to the trento-web UI. Depending on how you intend to connect to the console, a working hostname, FQDN, or an IP is required in `TRENTO_WEB_ORIGIN` for HTTPS. Otherwise websockets fail to connect, causing no real-time updates on the UI.



Note

Add `CHARTS_ENABLED=false` if Prometheus is not installed, or you do not want to use Trento's charts functionality.

```
docker run -d \
-p 4000:4000 \
--name trento-web \
--network trento-net \
--add-host "host.docker.internal:host-gateway" \
-e AMQP_URL=amqp://trento_user:trento_user_password@host.docker.internal/vhost \
-e ENABLE_ALERTING=false \
-e DATABASE_URL=ecto://trento_user:web_password@host.docker.internal/trento \
-e EVENTSTORE_URL=ecto://trento_user:web_password@host.docker.internal/trento_event_store \
-e PROMETHEUS_URL='http://host.docker.internal:9090' \
-e SECRET_KEY_BASE=$TRENTO_SECRET_KEY_BASE \
-e ACCESS_TOKEN_ENC_SECRET=$ACCESS_TOKEN_ENC_SECRET \
-e REFRESH_TOKEN_ENC_SECRET=$REFRESH_TOKEN_ENC_SECRET \
-e ADMIN_USER='admin' \
-e ADMIN_PASSWORD='test1234' \
-e ENABLE_API_KEY='true' \
-e TRENTO_WEB_ORIGIN='trento.example.com' \
--restart always \
--entrypoint /bin/sh \
registry.suse.com/trento/trento-web:latest \
-c "/app/bin/trento eval 'Trento.Release.init()' && /app/bin/trento start"
```

Email alerting are disabled by default, as described in [enabling alerting \(https://github.com/trento-project/web/blob/main/guides/alerting/alerting.md#enabling-alerting\)](https://github.com/trento-project/web/blob/main/guides/alerting/alerting.md#enabling-alerting) guide. Enable alerting by setting `ENABLE_ALERTING` env to `true`. Additional required variables are: `[ALERT_SENDER,ALERT_RECIPIENT,SMTP_SERVER,SMTP_PORT,SMTP_USER,SMTP_PASSWORD]` All other settings should remain at their default.

Example:

```
docker run -d \  
  
...[other settings]...  
  
-e ENABLE_ALERTING=true \  
-e ALERT_SENDER=<<SENDER_EMAIL_ADDRESS>> \  
-e ALERT_RECIPIENT=<<RECIPIENT_EMAIL_ADDRESS>> \  
-e SMTP_SERVER=<<SMTP_SERVER_ADDRESS>> \  
-e SMTP_PORT=<<SMTP_PORT>> \  
-e SMTP_USER=<<SMTP_USER>> \  
-e SMTP_PASSWORD=<<SMTP_PASSWORD>> \  
  
...[other settings]...
```

5. Check that everything is running as expected:

```
docker ps
```

Expected output:

CONTAINER ID	IMAGE	COMMAND	NAMES
CREATED	STATUS	PORTS	
8b44333aec39	registry.suse.com/trento/trento-web:2.2.0	"/bin/sh -c '/app/bi..."	
6 seconds ago	Up 5 seconds	0.0.0.0:4000->4000/tcp, :::4000->4000/tcp	trento-web
e859c07888ca	registry.suse.com/trento/trento-wanda:1.2.0	"/bin/sh -c '/app/bi..."	
18 seconds ago	Up 16 seconds	0.0.0.0:4001->4000/tcp, :::4001->4000/tcp	wanda

Both containers must run and listen on the specified ports.

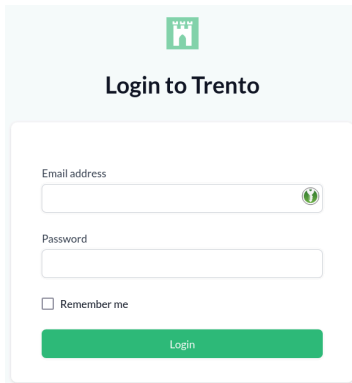
4.4 Automated deployment with Ansible

An automated installation of Trento Server using on RPM packages or Docker images can be performed with a Ansible playbook. For further information, refer to the official [Trento Ansible project \(https://github.com/trento-project/ansible\)](https://github.com/trento-project/ansible).

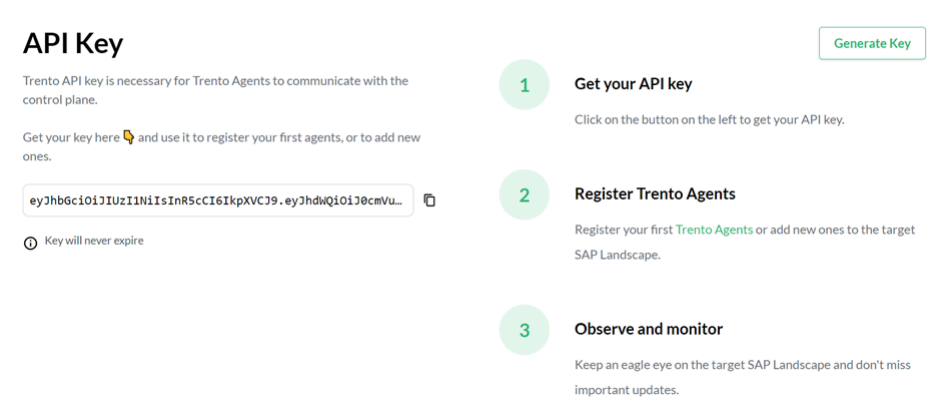
5 Installing Trento Agents

Before you can install a Trento Agent, you must obtain the API key of your Trento Server. Proceed as follows:

1. Open the URL of the Trento Web interface (http://TRENTO_SERVER_HOSTNAME). It prompts you for a user name and password:

A login form titled "Login to Trento" with a castle icon. It contains fields for "Email address" and "Password", a "Remember me" checkbox, and a green "Login" button.

2. Enter the credentials for the admin user (specified during installation of Trento Server).
3. Click *Login*.
4. When you are logged in, go to Settings:

A page titled "API Key" with a "Generate Key" button. It explains that the API key is necessary for Trento Agents to communicate with the control plane. It shows a generated key: "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdwQiOiJ0cmVud...". A note states "Key will never expire". On the right, a numbered list of steps: 1. Get your API key (Click on the button on the left to get your API key.), 2. Register Trento Agents (Register your first Trento Agents or add new ones to the target SAP Landscape.), 3. Observe and monitor (Keep an eagle eye on the target SAP Landscape and don't miss important updates.).

5. Click the *Copy* button to copy the key to the clipboard.

To install the Trento Agent on an SAP host and register it with the Trento Server, repeat the steps in *Procedure 2, "Installing Trento Agents"*:

PROCEDURE 2: INSTALLING TRENTO AGENTS

1. Install the package:

```
> sudo zypper ref
> sudo zypper install trento-agent
```

2. Open the configuration file `/etc/trento/agent.yaml` and uncomment (remove the `#` character) the entries for `facts-service-url`, `server-url` and `api-key`. Update the values as necessary:

- `facts-service-url`: the address of the AMQP RabbitMQ service used for communication with the checks engine (wanda). The correct value of this parameter depends on how Trento Server was deployed.

In a Kubernetes deployment, it is `amqp://trento:trento@TRENTO_SERVER_HOST-NAME:5672/`. If the default RabbitMQ username and password (`trento:trento`) were updated using Helm, the parameter must use a user-defined value.

In a systemd or containerized deployment, the correct value is `amqp://TRENTO_USER:TRENTO_USER_PASSWORD@TRENTO_SERVER_HOSTNAME:5672/vhost`. If `TRENTO_USER` and `TRENTO_USER_PASSWORD` have been replaced with custom values, you must use them.

- `server-url`: URL for the Trento Server (`http://TRENTO_SERVER_HOSTNAME`)
- `api-key`: the API key retrieved from the Web console

3. If SSL termination has been enabled on the server side, you can encrypt the communication from the agent to the server as follows:

- Provide an HTTPS URL instead of an HTTP one.
- Import the certificate from the Certificate Authority that has issued your Trento Server SSL certificate into the Trento Agent host as follows:

1. Copy the CA certificate in the PEM format to `/etc/pki/trust/anchors/`. If the CA certificate is in the CRT format, convert it to PEM using the following `openssl` command:

```
openssl x509 -in mycert.crt -out mycert.pem -outform PEM
```

2. Run the `update-ca-certificates` command.

4. Start the Trento Agent:

```
> sudo systemctl enable --now trento-agent
```

5. Check the status of the Trento Agent:

```
> sudo systemctl status trento-agent
● trento-agent.service - Trento Agent service
   Loaded: loaded (/usr/lib/systemd/system/trento-agent.service; enabled; vendor
  preset: disabled)
   Active: active (running) since Wed 2021-11-24 17:37:46 UTC; 4s ago
     Main PID: 22055 (trento)
        Tasks: 10
       CGroup: /system.slice/trento-agent.service
               └─22055 /usr/bin/trento agent start --consul-config-dir=/srv/consul/
                 consul.d
                   └─22220 /usr/bin/ruby.ruby2.5 /usr/sbin/SUSEConnect -s

[...]
```

6. Repeat this procedure on all SAP hosts that you want to monitor.

6 User management

Trento provides a local permission-based user management feature with optional multi-factor authentication. This feature enables segregation of duties in the Trento interface and ensures that only authorized users with the right permissions can access it.

User management actions are performed in the *Users* view in the left-hand side panel of the Trento Web console.

By default, a newly created user is granted display access rights except for the *Users* view. Where available, a user with default access can configure filters and pagination settings matching their preferences.

To perform protected actions, the user must have additional permissions added to their user profile. Below is the list of currently available permissions:

- `all:users`: grants full access to user management actions under the *Users* view
- `all:checks_selection`: grants check selection capabilities for any target in the registered environment for which checks are available

- all:checks_execution: grants check execution capabilities for any target in the registered environment for which checks are available and have been previously selected
- all:tags: allows creation and deletion of the available tags
- cleanup:all: allows triggering housekeeping actions on hosts where agents heartbeat is lost and SAP or HANA instances that are no longer found
- all:settings: grants changing capabilities on any system settings under the *Settings* view
- all:all: grants all the permissions above

Using the described permissions, it is possible to create the following types of users:

- **User managers:** users with all:users permissions
- **SAP Basis administrator with Trento display-only access:** users with default permissions
- **SAP Basis administrator with Trento configuration access:** users with all:checks_selection, all:tags and all:settings permissions
- **SAP Basis administrator with Trento operation access:** users with all:check_execution and cleanup:all permissions.

The default admin user created during the installation process is granted all:all permissions and cannot be modified or deleted. Use it only to create the first user manager (a user with all:users permissions who creates all the other required users). Once a user with all:users permissions is created, the default admin user must be treated as a fallback user in case all other access to the console is lost. If the password of the default admin user is lost, it can be reset by updating the Helm chart or the web component configuration, depending on which deployment method was used to install Trento Server.

User passwords, including the default admin user password, must follow the rules below:

- Password must contain at least 8 characters
- The same number or letter must not be repeated three or more times in a row (for example: 111 or aaa)
- Password must not contain four consecutive numbers or letters (for example: 1234, abcd or ABCD)

The *Create User* and *Edit User* views provide a built-in password generation button that allows user managers to easily generate secure and compliant passwords. The user manager must provide the user with their password through an authorized secure channel.

A user can reset their password in the *Profile* view. In this view, they can also update their name and email address as well as activate multi-factor authentication using an authenticator app. Multi-factor authentication increases the security of a user account by requesting a temporary second password or code when logging in the console. User managers can disable multi-factor authentication for any given user that has it enabled. However, user managers cannot enable multi-factor authentication on their behalf. The default admin user cannot enable its own multi-factor authentication.



Note: Security Tip for Multi-Factor Authentication

Since multi-factor authentication cannot be enabled for the default admin user, keeping its password safe is imperative. If the default admin user's password is compromised, reset it immediately by updating the Helm chart or the web component configuration, depending on which deployment method was used to install Trento Server.

User managers can enable and disable users. When a user logged in the console is disabled by a user admin, their session is terminated immediately.

7 Single Sign-On integration

Trento can be integrated for Single Sign-On with a third-party identity provider (IDP).



Note

Trento cannot start with multiple SSO options together, so only one can be chosen.

The following protocols are supported:

- OpenID Connect (OIDC)
- Open Authorization 2.0 (OAuth 2)
- Security Assertion Markup Language (SAML)

7.1 User Roles and Authentication

User authentication is entirely managed by the IDP, which is responsible for maintaining user accounts. A user, who does not exist on the IDP, is unable to access the Trento web console.

During the installation process, a default admin user is defined using the `ADMIN_USER` variable, which defaults to `admin`. If the authenticated user's IDP username matches this admin user's username, that user is automatically granted `all:all` permissions within Trento.

User permissions are entirely managed by Trento, they are not imported from the IDP. The abilities must be granted by some user with `all:all` or `all:users` abilities (admin user initially). This means that only basic user information is retrieved from the external IDP.

7.2 Using OpenID Connect

Trento integrates with an IDP that uses the OIDC protocol to authenticate users accessing the Trento web console.

By default, OIDC is disabled. You can enable OIDC when using RPM packages or using Docker images.

7.2.1 Enabling OpenID Connect when using kubernetes deployment

To enable OIDC when using kubernetes deployment with helm, proceed as follows:

1. Add the following variables to the previously documented helm installation command:

```
HELM_EXPERIMENTAL_OCI=1 helm ... \
  --set trento-web.oidc.enabled=true \
  --set trento-web.oidc.clientId=<OIDC_CLIENT_ID> \
  --set trento-web.oidc.clientSecret=<OIDC_CLIENT_SECRET> \
  --set trento-web.oidc.baseUrl=<OIDC_BASE_URL>
```

7.2.2 Enabling OpenID Connect when using RPM packages

To enable OIDC when using RPM packages, proceed as follows:

1. Open the file `/etc/trento/trento-web`.
2. Add the following environment variables to this file. Required variables are:

```
ENABLE_OIDC=true
```

```
OIDC_CLIENT_ID=<OIDC_CLIENT_ID>
OIDC_CLIENT_SECRET=<OIDC_CLIENT_SECRET>
OIDC_BASE_URL=<OIDC_BASE_URL>
```

3. Optionally, add the OIDC callback URL to the configuration. This can be useful if for some reason the default callback URL cannot be used, for example, if <http> is used instead of <https>. Use the next variable for that:

```
OIDC_CALLBACK_URL=<OIDC_CALLBACK_URL>
```

4. Restart the application.

7.2.3 Enabling OpenID Connect when using Docker images

To enable OIDC when using Docker images, proceed as follows:

1. If [trento-web](#) container is already running stop and delete the container before continuing. For that run:

```
docker stop trento-web
docker rm trento-web
```

2. Provide the following environment variables to the Docker container via the `-e` option:

```
docker run -d \
-p 4000:4000 \
--name trento-web \
--network trento-net \
--add-host "host.docker.internal:host-gateway" \

...[other settings]...

# Required:
-e ENABLE_OIDC=true \
-e OIDC_CLIENT_ID=<OIDC_CLIENT_ID> \
-e OIDC_CLIENT_SECRET=<OIDC_CLIENT_SECRET> \
-e OIDC_BASE_URL=<OIDC_BASE_URL> \

# Optional:
-e OIDC_CALLBACK_URL=<OIDC_CALLBACK_URL> \

...[other settings]...
```

7.2.4 Available variables for OpenID Connect

OIDC_CLIENT_ID

OIDC client id

OIDC_CLIENT_SECRET

OIDC client secret

OIDC_BASE_URL

OIDC base url

OIDC_CALLBACK_URL

OIDC callback url where the IDP is redirecting once the authentication is completed (default value: https://#{TRENTO_WEB_ORIGIN}/auth/oidc_callback)

7.3 Using OAuth 2.0

Trento integrates with an IDP that uses the OAuth 2 protocol to authenticate users accessing the Trento web console.

By default, OAuth 2.0 is disabled. You can enable OIDC when using RPM packages or using Docker images.

7.3.1 Enabling OAuth 2.0 when using kubernetes deployment

To enable OAuth 2.0 when using kubernetes deployment with helm, proceed as follows:

1. Add the following variables to the previously documented helm installation command:

```
HELM_EXPERIMENTAL_OCI=1 helm ... \  
  --set trento-web.oauth2.enabled=true \  
  --set trento-web.oauth2.clientId=<OAUTH2_CLIENT_ID> \  
  --set trento-web.oauth2.clientSecret=<OAUTH2_CLIENT_SECRET> \  
  --set trento-web.oauth2.baseUrl=<OAUTH2_BASE_URL> \  
  --set trento-web.oauth2.authorizeUrl=<OAUTH2_AUTHORIZE_URL> \  
  --set trento-web.oauth2.tokenUrl=<OAUTH2_TOKEN_URL> \  
  --set trento-web.oauth2.userUrl=<OAUTH2_USER_URL>
```

Additionally, the following optional values are available:

```
HELM_EXPERIMENTAL_OCI=1 helm ... \  

```

```
--set trento-web.oauth2.scopes=<OAUTH2_SCOPES>
```

7.3.2 Enabling OAuth 2.0 when using RPM packages

To enable OAuth 2.0 when using RPM packages, proceed as follows:

1. Open the file `/etc/trento/trento-web`.
2. Add the following environment variables to this file. Required variables are:

```
# Required:
ENABLE_OAUTH2=true
OAUTH2_CLIENT_ID=<OAUTH2_CLIENT_ID>
OAUTH2_CLIENT_SECRET=<OAUTH2_CLIENT_SECRET>
OAUTH2_BASE_URL=<OAUTH2_BASE_URL>
OAUTH2_AUTHORIZE_URL=<OAUTH2_AUTHORIZE_URL>
OAUTH2_TOKEN_URL=<OAUTH2_TOKEN_URL>
OAUTH2_USER_URL=<OAUTH2_USER_URL>

# Optional:
OAUTH2_SCOPES=<OAUTH2_SCOPES>
OAUTH2_CALLBACK_URL=<OAUTH2_CALLBACK_URL>
```

3. Restart the application.

7.3.3 Enabling OAuth 2.0 when using Docker images

To enable OAuth 2.0 when using Docker images, proceed as follows:

1. If `trento-web` container is already running stop and delete the container before continuing. For that run:

```
docker stop trento-web
docker rm trento-web
```

2. Use the following environment variables to the Docker container via the `-e` option:

```
docker run -d \
-p 4000:4000 \
--name trento-web \
--network trento-net \
--add-host "host.docker.internal:host-gateway" \
```

```
...[other settings]...

-e ENABLE_OAUTH2=true \
-e OAUTH2_CLIENT_ID=<OAUTH2_CLIENT_ID> \
-e OAUTH2_CLIENT_SECRET=<OAUTH2_CLIENT_SECRET> \
-e OAUTH2_BASE_URL=<OAUTH2_BASE_URL> \
-e OAUTH2_AUTHORIZE_URL=<OAUTH2_AUTHORIZE_URL> \
-e OAUTH2_TOKEN_URL=<OAUTH2_TOKEN_URL> \
-e OAUTH2_USER_URL=<OAUTH2_USER_URL> \

# Optional:
-e OAUTH2_SCOPES=<OAUTH2_SCOPES> \
-e OAUTH2_CALLBACK_URL=<OAUTH2_CALLBACK_URL> \

...[other settings]...
```

7.3.4 Available variables for OAuth 2.0

OAUTH2_CLIENT_ID

OAUTH2 client id

OAUTH2_CLIENT_SECRET

OAUTH2 client secret

OAUTH2_BASE_URL

OAUTH2 base url

OAUTH2_AUTHORIZE_URL

OAUTH2 authorization url

OAUTH2_TOKEN_URL

OAUTH2 token url

OAUTH2_USER_URL

OAUTH2 token url

OAUTH2_SCOPES

OAUTH2 scopes, used to define the user values sent to the SP. It must be adjusted depending on IDP provider requirements (default value: profile email)

OAUTH2_CALLBACK_URL

OAUTH2 callback url where the IDP is redirecting once the authentication is completed (default value: https://#{TRENTO_WEB_ORIGIN}/auth/oauth2_callback)

7.4 Using SAML

Trento integrates with an IDP that uses the SAML protocol to authenticate users accessing the Trento web console. Trento will behave as a Service Provider (SP) in this case.

Commonly, SAML protocol messages are signed with SSL. This is optional using Trento, and the signing is not required (even though it is recommended). If the IDP signs the messages, and expect signed messages back, certificates used by the SP (Trento in this case) must be provided to the IDP, the public certificate file in this case.

To use an existing SAML IDP, follow the next instructions to met the specific requirements. You need:

1. Obtain metadata content from the IDP
2. Start Trento to generate the certificates and get them (SAML must be enabled for this)
3. Provide the generated certificate to the IDP
4. Configure SAML IDP and user profiles

See the following subsections for details.

7.4.1 Obtaining metadata content from the IDP

The `metadata.xml` file defines the agreement between SP and IDP during SAML communications. It is used to identify the SAML client as well. The content of this file must be provided to Trento. Options `SAML_METADATA_URL` and `SAML_METADATA_CONTENT` are available for that.

If the `SAML_METADATA_CONTENT` option is being used, the content of this variable must be updated with the IDP metadata as single line string. On the other hand, if `SAML_METADATA_URL` is used, the new metadata is automatically fetched when Trento starts. If neither of these steps are completed, communication will fail because the message signatures will not be recognized.

If the used IDP has the endpoint to provide the `metadata.xml` file content, prefer the variable `SAML_METADATA_URL`. Trento will automatically fetch metadata when started.

7.4.2 Getting certificates from Trento

Trento provides a certificates set created during the installation. Regardless of the installation mode, when Trento is installed the first time and SAML is enabled the certificates are created and the public certificate file content is available in the `https://#{TRENTO_WEB_ORIGIN}/api/public_keys` route.

Use the following command to get the certificate content:

```
curl https://#{TRENTO_WEB_ORIGIN}/api/public_keys
```

Copy the content of the certificate from there and provide it to the IDP. This way, the IDP will sign its messages and verify the messages received from Trento.



Note

To get the certificate using this route Trento must be configured to start with SAML enabled.

7.4.3 Configuring SAML IDP setup

Configure the existing IDP with the next minimum options to be able to connect with Trento as a Service Provider (SP).

7.4.3.1 Providing certificates

As commented previously, a set of certificates is needed to enable signed communication. Provide the certificate generated by Trento to the IDP (each IDP has a different way to do this). Make sure that the configured certificate is used for signing and encrypting messages.

7.4.3.2 Configuring SAML user profile

Users provided by the SAML installation must have some few mandatory attributes to login in Trento. The required attributes are: `username`, `email`, `firstName` and `lastName`. All of them are mandatory, even though their field names are configurable.

By default, Trento expects the `username`, `email`, `firstName` and `lastName` attribute names. All these 4 attribute names are configurable using the next environment variables, following the same order: `SAML_USERNAME_ATTR_NAME`, `SAML_EMAIL_ATTR_NAME`, `SAML_FIRSTNAME_ATTR_NAME` and `SAML_LASTNAME_ATTR_NAME`.

Both IDP and Trento must know how these 4 fields are mapped. To do this, follow the next instructions:

1. Add the attributes if they don't exist in the IDP user profile. If they already exist, don't change the attributes and keep their original values.
2. Configure Trento to use the IDP attribute field names. To do this, set the `SAML_USERNAME_ATTR_NAME`, `SAML_EMAIL_ATTR_NAME`, `SAML_FIRSTNAME_ATTR_NAME` and `SAML_LASTNAME_ATTR_NAME` environment values with the values configured in the IDP. For example, if the IDP user profile username is defined as `attr:username` use `SAML_USERNAME_ATTR_NAME=attr:username`.

7.4.3.3 Checking SAML redirect URI

After a successful login, the IDP redirects the user's session back to Trento and redirected at `https://trento.example.com/sso/sp/consume/saml`. To ensure seamless SSO, this URI must be configured as valid within the IDP.

7.4.4 Enabling SAML when using kubernetes deployment

To enable SAML when using kubernetes deployment with helm, proceed as follows:

1. Add the following variables to the previously documented helm installation command:

```
HELM_EXPERIMENTAL_OCI=1 helm ... \
  --set trento-web.saml.enabled=true \
  --set trento-web.saml.idpId=<SAML_IDP_ID> \
  --set trento-web.saml.spId=<SAML_SP_ID> \
  --set trento-web.saml.metadataUrl=<SAML_METADATA_URL>
```

To use the `SAML_METADATA_CONTENT` option rather than `SAML_METADATA_URL` use:

```
HELM_EXPERIMENTAL_OCI=1 helm ... \
  --set trento-web.saml.enabled=true \
  --set trento-web.saml.idpId=<SAML_IDP_ID> \
  --set trento-web.saml.spId=<SAML_SP_ID> \
  --set trento-web.saml.metadataContent=<SAML_METADATA_CONTENT>
```

Additionally, the following optional values are available:

```
HELM_EXPERIMENTAL_OCI=1 helm ... \
```

```
--set trento-web.saml.idpNameIdFormat=<SAML_IDP_NAMEID_FORMAT> \
--set trento-web.saml.spDir=<SAML_SP_DIR> \
--set trento-web.saml.spEntityId=<SAML_SP_ENTITY_ID> \
--set trento-web.saml.spContactName=<SAML_SP_CONTACT_NAME> \
--set trento-web.saml.spContactEmail=<SAML_SP_CONTACT_EMAIL> \
--set trento-web.saml.spOrgName=<SAML_SP_ORG_NAME> \
--set trento-web.saml.spOrgDisplayName=<SAML_SP_ORG_DISPLAYNAME> \
--set trento-web.saml.spOrgUrl=<SAML_SP_ORG_URL> \
--set trento-web.saml.usernameAttrName=<SAML_USERNAME_ATTR_NAME> \
--set trento-web.saml.emailAttrName=<SAML_EMAIL_ATTR_NAME> \
--set trento-web.saml.firstNameAttrName=<SAML_FIRSTNAME_ATTR_NAME> \
--set trento-web.saml.lastNameAttrName=<SAML_LASTNAME_ATTR_NAME> \
--set trento-web.saml.signRequests=<SAML_SIGN_REQUESTS> \
--set trento-web.saml.signMetadata=<SAML_SIGN_METADATA> \
--set trento-web.saml.signedAssertion=<SAML_SIGNED_ASSERTION> \
--set trento-web.saml.signedEnvelopes=<SAML_SIGNED_ENVELOPES>
```

7.4.5 Enabling SAML when using RPM packages

To enable SAML when using RPM packages, proceed as follows:

1. Open the file `/etc/trento/trento-web`.
2. Add the following environment variables to this file. Required variables are:

```
# Required:
ENABLE_SAML=true
SAML_IDP_ID=<SAML_IDP_ID>
SAML_SP_ID=<SAML_SP_ID>
# Only SAML_METADATA_URL or SAML_METADATA_CONTENT must be provided
SAML_METADATA_URL=<SAML_METADATA_URL>
SAML_METADATA_CONTENT=<SAML_METADATA_CONTENT>

# Optional:
SAML_IDP_NAMEID_FORMAT=<SAML_IDP_NAMEID_FORMAT>
SAML_SP_DIR=<SAML_SP_DIR>
SAML_SP_ENTITY_ID=<SAML_SP_ENTITY_ID>
SAML_SP_CONTACT_NAME=<SAML_SP_CONTACT_NAME>
SAML_SP_CONTACT_EMAIL=<SAML_SP_CONTACT_EMAIL>
SAML_SP_ORG_NAME=<SAML_SP_ORG_NAME>
SAML_SP_ORG_DISPLAYNAME=<SAML_SP_ORG_DISPLAYNAME>
SAML_SP_ORG_URL=<SAML_SP_ORG_URL>
SAML_USERNAME_ATTR_NAME=<SAML_USERNAME_ATTR_NAME>
SAML_EMAIL_ATTR_NAME=<SAML_EMAIL_ATTR_NAME>
SAML_FIRSTNAME_ATTR_NAME=<SAML_FIRSTNAME_ATTR_NAME>
SAML_LASTNAME_ATTR_NAME=<SAML_LASTNAME_ATTR_NAME>
```

```
SAML_SIGN_REQUESTS=<SAML_SIGN_REQUESTS>
SAML_SIGN_METADATA=<SAML_SIGN_METADATA>
SAML_SIGNED_ASSERTION=<SAML_SIGNED_ASSERTION>
SAML_SIGNED_ENVELOPES=<SAML_SIGNED_ENVELOPES>
```

3. Restart the application.

7.4.6 Enabling SAML when using Docker images

To enable SAML when using Docker images, proceed as follows:

1. If `trento-web` container is already running stop and delete the container before continuing. For that run:

```
docker stop trento-web
docker rm trento-web
```

2. Use the following environment variables to the Docker container via the `-e` option:

```
docker run -d \
-p 4000:4000 \
--name trento-web \
--network trento-net \
--add-host "host.docker.internal:host-gateway" \

...[other settings]...

-e ENABLE_SAML=true
-e SAML_IDP_ID=<SAML_IDP_ID> \
-e SAML_SP_ID=<SAML_SP_ID> \
# Only SAML_METADATA_URL or SAML_METADATA_CONTENT must be provided
-e SAML_METADATA_URL=<SAML_METADATA_URL> \
-e SAML_METADATA_CONTENT=<SAML_METADATA_CONTENT> \

# Optional:
-e SAML_IDP_NAMEID_FORMAT=<SAML_IDP_NAMEID_FORMAT> \
-e SAML_SP_DIR=<SAML_SP_DIR> \
-e SAML_SP_ENTITY_ID=<SAML_SP_ENTITY_ID> \
-e SAML_SP_CONTACT_NAME=<SAML_SP_CONTACT_NAME> \
-e SAML_SP_CONTACT_EMAIL=<SAML_SP_CONTACT_EMAIL> \
-e SAML_SP_ORG_NAME=<SAML_SP_ORG_NAME> \
-e SAML_SP_ORG_DISPLAYNAME=<SAML_SP_ORG_DISPLAYNAME> \
-e SAML_SP_ORG_URL=<SAML_SP_ORG_URL> \
-e SAML_USERNAME_ATTR_NAME=<SAML_USERNAME_ATTR_NAME> \
-e SAML_EMAIL_ATTR_NAME=<SAML_EMAIL_ATTR_NAME> \
```

```
-e SAML_FIRSTNAME_ATTR_NAME=<SAML_FIRSTNAME_ATTR_NAME> \  
-e SAML_LASTNAME_ATTR_NAME=<SAML_LASTNAME_ATTR_NAME> \  
-e SAML_SIGN_REQUESTS=<SAML_SIGN_REQUESTS> \  
-e SAML_SIGN_METADATA=<SAML_SIGN_METADATA> \  
-e SAML_SIGNED_ASSERTION=<SAML_SIGNED_ASSERTION> \  
-e SAML_SIGNED_ENVELOPES=<SAML_SIGNED_ENVELOPES> \  
  
...[other settings]...
```

7.4.7 Available variables for SAML

SAML_IDP_ID

SAML IDP id

SAML_SP_ID

SAML SP id

SAML_METADATA_URL

URL to retrieve the SAML metadata xml file. One of SAML_METADATA_URL or SAML_METADATA_CONTENT is required

SAML_METADATA_CONTENT

One line string containing the SAML metadata xml file content (SAML_METADATA_URL has precedence over this)

SAML_IDP_NAMEID_FORMAT

SAML IDP name id format, used to interpret the attribute name. Whole urn string must be used (default value: urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified)

SAML_SP_DIR

SAML SP directory, where SP specific required files (such as certificates and metadata file) are placed (default value: /etc/trento/saml)

SAML_SP_ENTITY_ID

SAML SP entity id. If it is not given, value from the metadata.xml file is used

SAML_SP_CONTACT_NAME

SAML SP contact name (default value: Trento SP Admin)

SAML_SP_CONTACT_EMAIL

SAML SP contact email (default value: admin@trento.suse.com)

SAML_SP_ORG_NAME

SAML SP organization name (default value: Trento SP)

SAML_SP_ORG_DISPLAYNAME

SAML SP organization display name (default value: SAML SP build with Trento)

SAML_SP_ORG_URL

SAML SP organization url (default value: https://www.trento-project.io/)

SAML_USERNAME_ATTR_NAME

SAML user profile "username" attribute field name. This attribute must exist in the IDP user (default value: username)

SAML_EMAIL_ATTR_NAME

SAML user profile "email" attribute field name. This attribute must exist in the IDP user (default value: email)

SAML_FIRSTNAME_ATTR_NAME

SAML user profile "first name" attribute field name. This attribute must exist in the IDP user (default value: firstName)

SAML_LASTNAME_ATTR_NAME

SAML user profile "last name" attribute field name. This attribute must exist in the IDP user (default value: lastName)

SAML_SIGN_REQUESTS

Sign SAML requests in the SP side (default value: true)

SAML_SIGN_METADATA

Sign SAML metadata documents in the SP side (default value: true)

SAML_SIGNED_ASSERTION

Require to receive SAML assertion signed from the IDP. Set to false if the IDP doesn't sign the assertion (default value: true)

SAML_SIGNED_ENVELOPES

Require to receive SAML envelopes signed from the IDP. Set to false if the IDP doesn't sign the envelopes (default value: true)

8 Activity Log

Trento collects system events and user actions in the Activity Log. It can be accessed from the left-hand side panel of the Trento console.

Each entry in the Activity Log includes the following:

- A timestamp: the day and time (server time, not browser time) the system event or the user action occurred
- A message: type of the occurred event or user action
- The user that triggered the event or performed the action. User system is used for events triggered by the system itself.

The right chevron icon in the activity log entry opens a modal with the activity metadata.

The Activity Log allows to filter by the type of event or user action, commonly referred to as resource type, and by the user that triggered the event or performed the action. Only active users are available for filtering. The Activity Log also allows to filter out entries that are newer and/or older than a specific date and time (server time).

Once a filter has been set, the user must click Apply to filter out the undesired entries and Reset to remove all the filters.

Entries related to user management can only be displayed by users that have the all:all or all:users permissions. This includes:

- Login attempts
- User creations
- User modifications
- User deletions
- Profile updates.

Entries in the Activity Log are sorted from newer to older. By clicking Refresh, the user can update the Activity Log view with entries generated since they accessed the view, or after the last refresh.

The pagination features at the bottom of the Activity Log allow the user to specify the number of entries to display in a page, scroll back and forth through the different pages of the view, and jump directly to the last page and back to first one.

The default retention time for entries in the Activity Log is one month. This can be changed in the Activity Log section under Settings. Changing the retention time requires the `all:settings` permissions. Entries expired after the specified retention time are deleted every day at midnight (server time).

9 Performing configuration checks

One of Trento's main features is to provide configuration checks that ensure your infrastructure setup adheres to our Best Practices, or other vendor's Best Practices, and does not drift away in time. Configuration checks are available for HANA clusters, ASCS/ERS clusters and hosts. The following procedure is specific to a HANA cluster. The procedure for an ASCS/ERS cluster or a host would be exactly the same, only starting from the corresponding Details view:

PROCEDURE 3: PERFORMING CONFIGURATION CHECKS ON A HANA CLUSTER

1. Log in to Trento
2. In the left panel, click *Cluster*.
3. In the list, search for an SAP HANA cluster.
4. Click the respective cluster name in the *Name* column. The *Details* view opens.

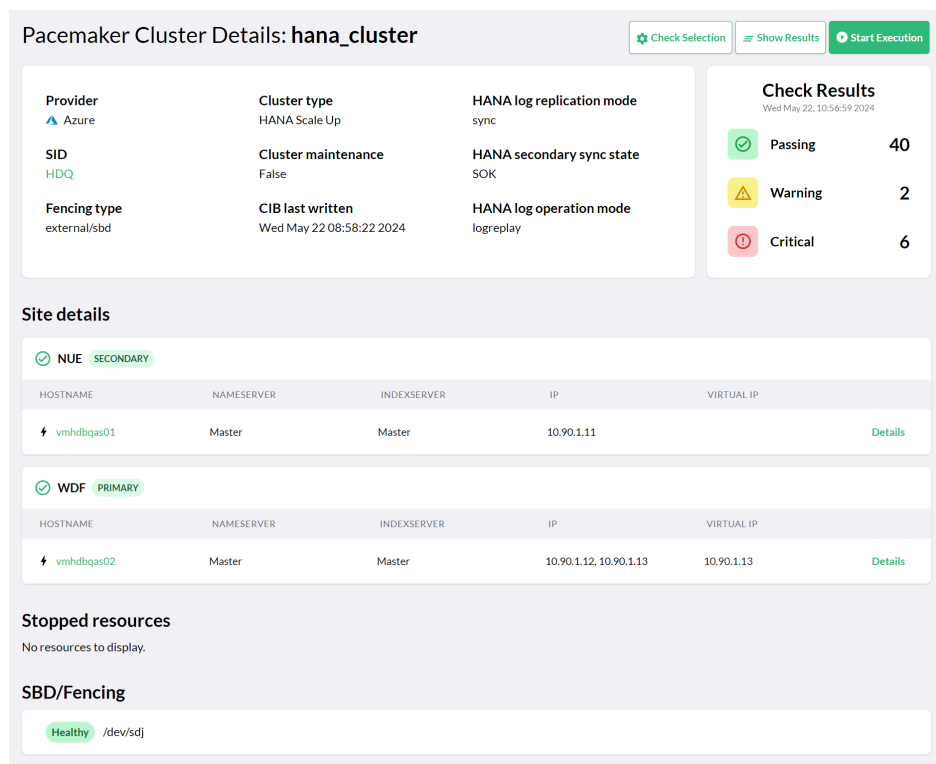


FIGURE 2: PACEMAKER CLUSTER DETAILS

- Click the *Settings* button to change the cluster settings of the respective cluster. For checks to be executed, a checks selection must be made. Select the checks to be executed and click the button *Select Checks for Execution*. See Figure 3, “Pacemaker Cluster Settings—Checks Selection”:

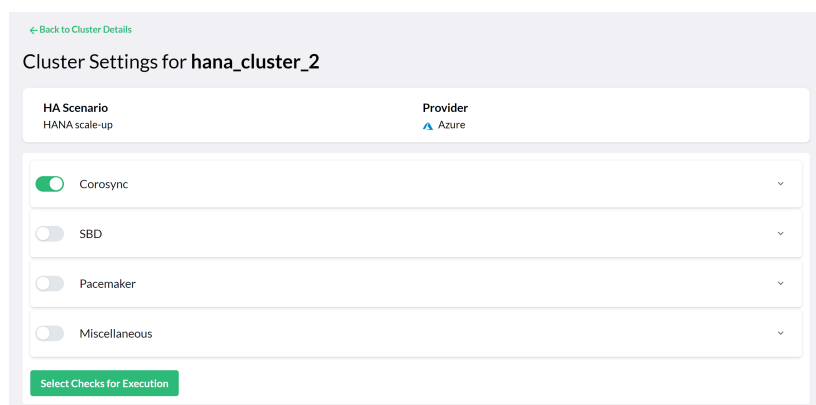


FIGURE 3: PACEMAKER CLUSTER SETTINGS—CHECKS SELECTION

- At this moment, you can either wait for Trento to execute the selected checks or trigger an execution immediately by clicking the button that has appeared in the Checks Selection tab.
- Investigate the result in the Checks Results view. Each row in this view shows you a check ID, a short description of the check and the check execution result. By clicking on a row, a section will open providing information about the execution on each node of the cluster. See *Figure 4, "Check results for a cluster"*:

← Back to Cluster Details

Checks Results for cluster hana_cluster


critical







✕

HA Scenario

HANA scale-up

Provider

 Azure

ID	DESCRIPTION	RESULT						
156F64	Corosync token timeout is set to expected value	<div>ⓘ</div>						
<table><tr><th>TARGET</th><th>EXPECTATIONS</th></tr><tr><td><div> vmhdbqas02</div></td><td>0/1 Expectations met.</td></tr><tr><td><div> vmhdbqas01</div></td><td>0/1 Expectations met.</td></tr></table>			TARGET	EXPECTATIONS	<div> vmhdbqas02</div>	0/1 Expectations met.	<div> vmhdbqas01</div>	0/1 Expectations met.
TARGET	EXPECTATIONS							
<div> vmhdbqas02</div>	0/1 Expectations met.							
<div> vmhdbqas01</div>	0/1 Expectations met.							

B3DA7E	<div>Premium</div>	Cluster resource-stickiness and migration-threshold are properly configured	<div>ⓘ</div>
53D035		Corosync is running with token timeout set to the recommended value	<div>ⓘ</div>

FIGURE 4: CHECK RESULTS FOR A CLUSTER

The result of a check execution can be passing, warning or critical:

- Passing* means that the checked configuration meets the recommendation.
- Warning* means that the recommendation is not met but the configuration is not critical for the proper running of the cluster.
- Critical* means that either the execution itself errored (for example, a timeout) or the recommendation is not met and is critical for the well-being of the cluster.

Use the filter to reduce the list to only show, for example, critical results.

- Click a check's link to open the *Details* view of this check. This shows you an abstract and how to remedy the problem. The *References* section contains links to the documentation of the different vendors to provide more context when necessary. Close the view with the **Esc** key or click outside of the view.

For each not-met expectation, there will be a detailed view providing information about it: what facts were gathered, what values were expected and what was the result of the evaluation. This will help users understand better why a certain configuration check is failing:

Corosync `token` timeout is set to expected value		
Check ID 156f64	Host vmhdbqas02	Provider Azure
Evaluation Results token_timeout Critical Corosync `token` timeout value was expected to be '30000' but configured value is '20000'		
Expected Values expected_token_timeout 30000		
Gathered Facts corosync_token_timeout 20000		

FIGURE 5: NON-MET EXPECTATION DETAIL VIEW

Once a check selection for a given cluster has been made, Trento executes them automatically every five minutes, and results are updated accordingly. A check execution result icon spinning means that an execution is running.

10 Using Trento Web console

The left sidebar in the Trento Web console contains the following entries:

- **Dashboard.** Determine at a glance the health status of your SAP environment.
- **Hosts.** Overview of all registered hosts running the Trento Agent.
- **Clusters.** Overview of all discovered Pacemaker clusters.
- **SAP Systems.** Overview of all discovered SAP Systems; identified by the corresponding system IDs.
- **HANA Databases.** Overview of all discovered SAP HANA databases; identified by the corresponding system IDs.
- **Checks catalog.** Overview of the catalog of configuration checks that Trento may perform on the different targets (hosts or clusters), cluster types (HANA scale up, HANA scale out or ASCS/ERS) and supported platforms (Azure, AWS, GCP, Nutanix, on-premises/KVM or VMware).

- **Settings.** Lets you retrieve and operate the API key for this particular installation, which is required for the Trento Agent configuration, as well as manage the connection data for a SUSE Manager instance and the retention time for the entries in the activity log.
- **About.** Shows the Trento flavor, the current server version, a link to the GitHub repository of the Trento Web component, and the number of registered SUSE Linux Enterprise Server for SAP Applications subscriptions that has been discovered.

10.1 Getting the global health state

The dashboard allows you to determine at a glance the health status of your SAP environment. It is the home page of the Trento Web console, and you can go back to it any time by clicking on *Dashboard* in the left sidebar.

The health status of a registered SAP system is the compound of its health status at three different layers representing the SAP architecture:

- **Hosts:** it reflects the heartbeat of the Trento Agent and, when applicable, the compliance status.
- **Pacemaker Clusters:** the status here is based on the running status of the cluster and the results of the configuration checks.
- **Database:** it collects the status of the HANA instances as returned by sapcontrol.
- **Application instances:** it summarizes the status of the application instances as returned by sapcontrol.

Complementing the operating system layer, you also have information about the health status of the HA components, whenever they exit:

- **Database cluster:** the status here is based on the running status of the database cluster and the results of the selected configuration checks.
- **Application cluster:** the status here is based on the running status of the ASCS/ERS cluster and, eventually, the results of the selected configuration checks.

The dashboard groups systems in three different health boxes (see *Figure 6, “Dashboard with the global health state”*):

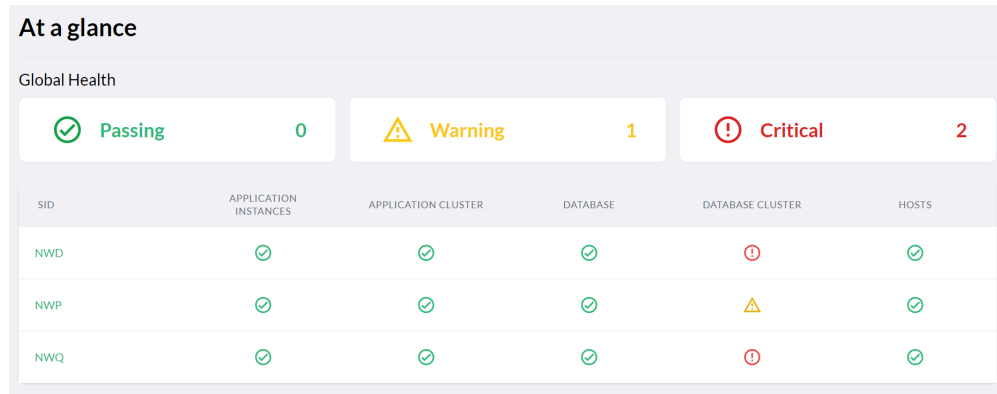


FIGURE 6: DASHBOARD WITH THE GLOBAL HEALTH STATE

THE THREE DIFFERENT HEALTH STATES

Passing

It shows the number of systems with all layers in passing (green) status.

Warning

It shows the number of systems with at least one layer in warning (yellow) status and the rest in passing (green) status.

Critical

It shows the number of systems with at least one layer in critical (red) status.

The health boxes in the dashboard are clickable. By clicking on one particular box, you filter the dashboard by systems with the corresponding health status. In large SAP environments, this feature facilitates the SAP administrator knowing which systems are in a given status.

The icons representing the health summary of a particular layer contain links to the views in the Trento console that can help determine where an issue is coming from:

- **Hosts health icon:** Link to the Hosts overview, filtered by SID equal to the SAPSID and the DBSID of the corresponding SAP system.
- **Database cluster health icon:** Link to the corresponding SAP HANA Cluster Details view.
- **Database health icon:** Link to the corresponding HANA Database Details view.
- **Application cluster health icon:** Link to the corresponding ASCS/ERS Cluster Details view.
- **Application Instances health icon:** Link to the corresponding SAP System Details view.

A grey status is returned when either a component does not exist, or it is stopped (as returned by `sapcontrol`), or its status is unknown (for instance, if a command to determine the status fails). Grey statuses are not yet counted in the calculation of the global health status.

10.2 Viewing the status

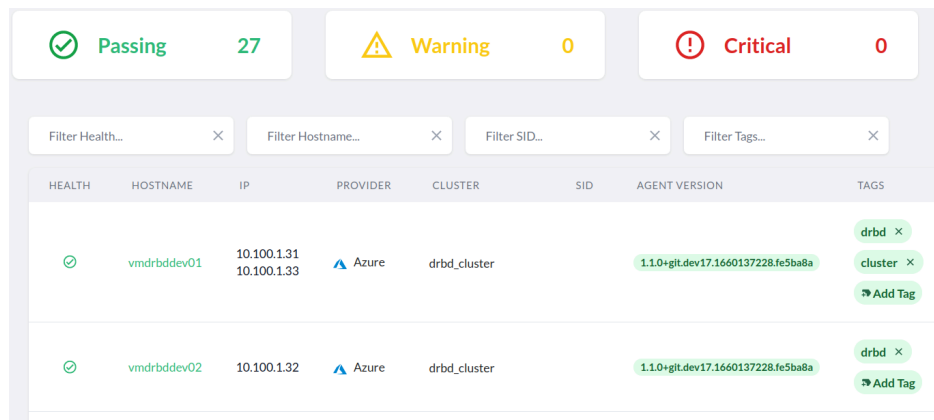
Getting to know the status of a system is an important task for every administrator. The status allows you to see if some of your systems need to be examined further.

The following subsection gives you an overview of specific parts of your SAP Landscape to show their state. Each status site shows you an overview of the health states (see details in *The three different health states*).

10.3 Viewing the status of hosts

To display the lists of registered hosts and their details, proceed as follows:

1. Log in to the Trento Web console.
2. Click the *Hosts* entry in the left sidebar to show a summary of the state for all hosts (see *Figure 7, "Hosts entry"*).



The screenshot displays the 'Hosts' entry in the Trento Web console. At the top, there are three summary cards: 'Passing' with 27 hosts, 'Warning' with 0 hosts, and 'Critical' with 0 hosts. Below these are four filter buttons: 'Filter Health...', 'Filter Hostname...', 'Filter SID...', and 'Filter Tags...'. The main table lists host details with columns: HEALTH, HOSTNAME, IP, PROVIDER, CLUSTER, SID, AGENT VERSION, and TAGS. Two hosts are visible, both with a 'Passing' status.

HEALTH	HOSTNAME	IP	PROVIDER	CLUSTER	SID	AGENT VERSION	TAGS
✓	vmdrbddev01	10.100.1.31 10.100.1.33	Azure	drbd_cluster		1.1.0+git.dev17.1660137228.fe5ba8a	drbd × cluster × Add Tag
✓	vmdrbddev02	10.100.1.32	Azure	drbd_cluster		1.1.0+git.dev17.1660137228.fe5ba8a	drbd × Add Tag

FIGURE 7: HOSTS ENTRY

3. To investigate the specific host details, click the host name in the respective column to open the corresponding *Host details* view. If the list is too big, reduce the list by providing filters.

Clicking on a host name opens the corresponding *Host details* view. This view provides the following:

HOST DETAILS VIEW

- *Hosts details* section: it shows the status of both the Trento Agent and the Node Exporter and provides the host name, the cluster name (when applicable), the Trento Agent version and the host IP addresses.
- *saptune summary* section: saptune is a solution that comes with SUSE Linux Enterprise Server for SAP Applications and allows SAP administrators to ensure that their SAP hosts are properly configured to run the corresponding SAP workloads. The integration of saptune in the Trento console gives the SAP administrator visibility over the tool even when they are not working at operating system level. The integration supports saptune 3.1.0 and higher, and includes the addition of the host tuning status in the aggregated health status of the host.

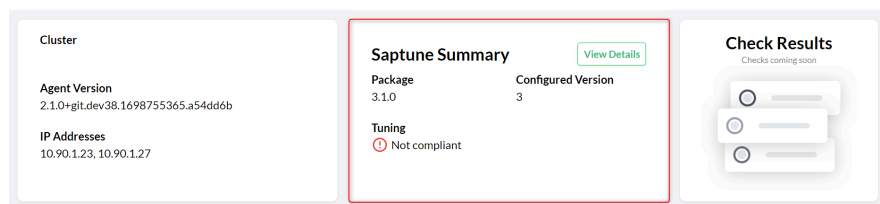


FIGURE 8: SAPTUNE SUMMARY SECTION

If an SAP workload is running on the host but no saptune or a version lower than 3.1.0 is installed, a warning is added to the aggregated health status of the host. When saptune version 3.1.0 or higher, is installed, a details view is also available showing detailed information about the saptune status:

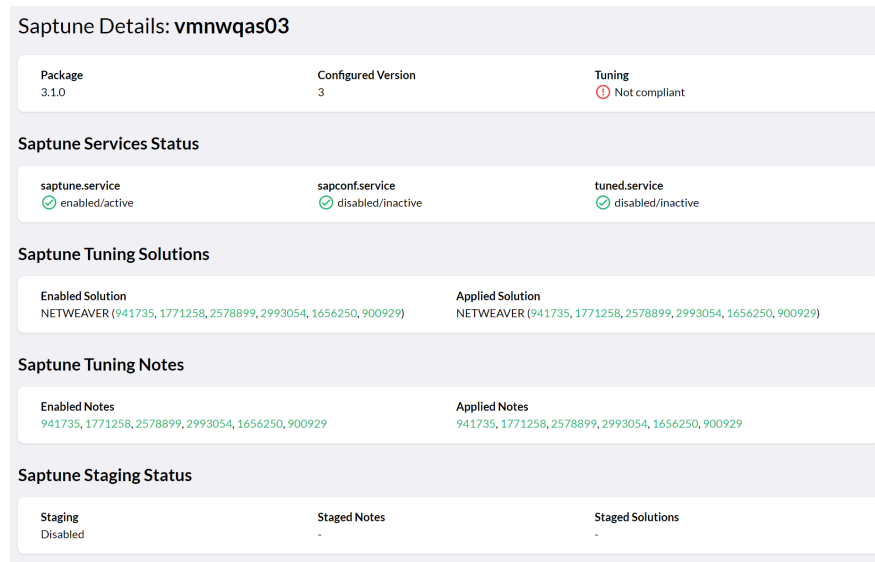
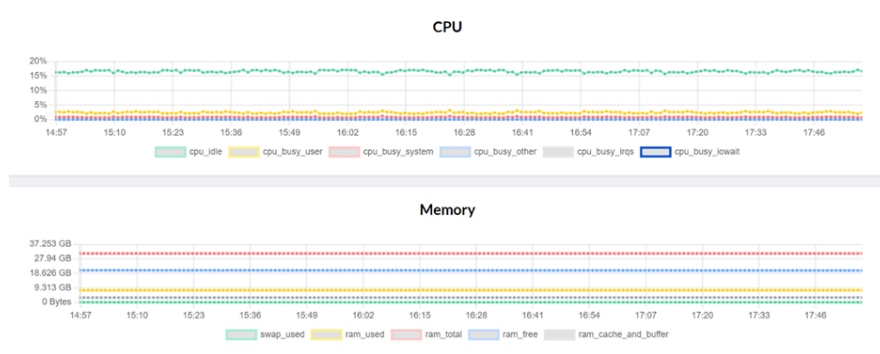


FIGURE 9: SAPTUNE DETAILS VIEW

- **Checks results summary section:** this section shows a summary of the checks execution results for this particular host.
- **Available software updates sections:** when settings for SUSE Manager are maintained and the host is managed by the SUMA instance for which connection data has been provided, this section shows a summary of the available patches and upgradable packages for this particular host. Refer to [Section 13, "Integration with SUSE Manager"](#) for further details.
- **Monitoring dashboard:** it shows the CPU and Memory usage for the specific hosts.



- *Provider details* section: when applicable, it shows the name of the cloud provider, the name of the virtual machine, the name of the resource group it belongs to, the location, the size of the virtual machine, and other information.
- *SAP instances* section: when applicable, it lists the ID, SID, type, features, and instance number of any SAP instance running on the host (SAP NetWeaver or SAP HANA).
- *SUSE subscription details* section: it lists the different components or modules that are part of the subscription, and for each one of them, it provides the architecture, the version and type, the registration and subscription status as well as the start and end dates of the subscription.

10.4 Viewing the Pacemaker cluster status

To display a list of all available Pacemaker clusters and their details, proceed as follows:

1. Log in to the Trento Web console.
2. Click the *Clusters* entry in the left sidebar to show a summary of the state for all Pacemaker clusters (see [Figure 10, "Pacemaker clusters"](#)).

<div> <div> ✓ </div> <div> <div>Passing</div> <div>1</div> </div> </div>		<div> <div> ⚠ </div> <div> <div>Warning</div> <div>0</div> </div> </div>		<div> <div> ❗ </div> <div> <div>Critical</div> <div>1</div> </div> </div>	
Filter Health...	Filter Name...	Filter SID...	Filter Type...	Filter Tags...	
HEALTH	NAME	SID	HOSTS	RESOURCES	TAGS
●	drbd_cluster		2	9	Unknown Add Tag
⌚	hana_cluster	HDQ	2	7	HANA Scale Up Add Tag
✓	netweaver_cluster	NWQ	2	9	ASCS/ERS Add Tag

FIGURE 10: PACEMAKER CLUSTERS

3. To investigate the specific Pacemaker cluster details, click the cluster name in the respective column to open the corresponding *Pacemaker cluster details* view. If the list is too big, reduce the list by providing filters.

The detail views of a HANA cluster and an ASCS/ERS cluster are different:

HANA CLUSTER DETAILS VIEW

- The *Settings*, *Show Results*, and *Start Execution* buttons. These buttons are used to enable or disable checks and to start them. If you want to perform specific checks, proceed with *Step 5* of *Procedure 3, "Performing configuration checks on a HANA Cluster"*.
- Top section: it displays the cloud provider, the cluster type, the HANA log replication mode, the DBSID, the cluster maintenance status, the HANA secondary sync state, the fencing type, when the CIB was last written, and the HANA log operation mode.
- *Check Result* section: with a summary of the health of the clusters based on the runtime status and the selected configuration checks for each one of them (passed, warning and critical).
- *Pacemaker Site details* section: this section is split in three subsections. One for each HANA site and another ones for cluster nodes without a HANA workload. For example, a majority maker in the case of a HANA scale out cluster. Each HANA site subsection will inform about the site role (Primary or Secondary or Failed) and will list the different nodes in the site. Each node entry will display the node status (Online or Maintenance or Other), the roles of the nameserver and indexserver services in

that node, the local IPs and any assigned virtual IP address. If you click the *Details* button, you can view the attributes of that node, the resources running on it and their statuses. Close this view with the **Esc** key.

- *Stopped resources* section: with a summary of resources which have been stopped on the cluster.
- *SBD/Fencing* section: with the status of each SBD device when applicable.

ASCS/ERS CLUSTER DETAILS VIEW

- A top section on the left showing the cloud provider, the cluster type, fencing type, when the CIB was last written and the cluster maintenance status.
- Another top centered section showing the SAP SID, the Enqueue Server version, whether the ASCS and ERS are running on different hosts or not and whether the instance filesystems are resource based or not. This is a multi-tab section. When multiple systems share the same cluster, there will be a tab for each system in the cluster and you can scroll left and right to go through the different systems.
- A *Check Result* section, showing a summary of the results of the last check execution, when applicable.
- A *Node details* section showing the following for each node in the cluster: the node status (Online or Maintenance or Other), the host name, the role of the node in the cluster, the assigned virtual IP address and, in case of resource managed filesystems, the full mounting path. If you click on *Details*, you can view the attributes and resources associated to that particular node. Close this view with the **Esc** key. This section is system specific. It will show the information corresponding to the system selected in the multi-tab section above.
- A *Stopped resources* section: with a summary of resources which have been stopped on the cluster.
- *SBD/Fencing* section: with the status of each SBD device when applicable.

10.5 Viewing the SAP Systems status

To display a list of all available SAP Systems and their details, proceed as follows:

1. Log in to the Trento Web console.

- To investigate the specific SAP Systems details, click the *SAP Systems* entry in the left sidebar to show a summary of the state for all SAP Systems (see [Figure 11, “SAP Systems”](#)).

✓

Passing

3

⚠

Warning

0

❗

Critical

0

Filter Health...

Filter SID...

Filter Tags...

HEALTH	SID	ATTACHED RDBMS	TENANT	DB ADDRESS	ENSA VERSION	TAGS
✓	NWD	HDD	HDD	10.100.1.13	ENSA1	<div>PRD × S4HANA × Add Tag</div>
✓	NWP	HDP	HDP	10.80.1.13	ENSA1	<div>Add Tag</div>
✓	NWQ	HDQ	HDQ	10.90.1.13	ENSA1	<div>Add Tag</div>

1

FIGURE 11: SAP SYSTEMS


- Decide which part you want to examine. You can view the details of the *SID* column of an entry, or you go with *Attached RDBMS* to see the details of the database. If the list is too big, reduce the list by providing filters.

If you click on an entry in the *SID* column, the *SAP System Details* view opens. This view provides the following:

SAP SYSTEM DETAILS

- The *Name* and *Type* of this SAP System.
- Layout* section: for each instance, virtual host name, instance number, features (processes), HTTP and HTTPS ports, start priority, and SAPControl status.
- Hosts* section: with the host name, the IP address, the cloud provider (when applicable), the cluster name (when applicable), and the Trento Agent version for each listed host. When you click the host name, it takes you to the corresponding [Host details view](#).

SAP System Details

Name NWD	Type Application server	ENSA version ENSA1					
-------------	----------------------------	-----------------------	--	--	--	--	---

Layout

HOSTNAME	INSTANCE NR	FEATURES	HTTP PORT	HTTPS PORT	START PRIO	STATUS
sapnwdas	00	MESSAGESEVER ENQUE	50013	50014	1	SAPControl: ● Green
sapnwdpas	01	ABAP GATEWAY ICMAN IGS	50113	50114	3	SAPControl: ● Green
sapnwdas1	02	ABAP GATEWAY ICMAN IGS	50213	50214	3	SAPControl: ● Green
sapnwdcr	10	ENQREP	51013	51014	0.5	SAPControl: ● Green

Hosts





HOSTNAME	IP	PROVIDER	CLUSTER	AGENT VERSION
vmnwdev01	10.100.1.21 10.100.1.25	 Azure	netweaver_cluster	2.1.0
vmnwdev03	10.100.1.23 10.100.1.27	 Azure		2.1.0
vmnwdev04	10.100.1.24 10.100.1.28	 Azure		2.1.0
vmnwdev02	10.100.1.22 10.100.1.26	 Azure	netweaver_cluster	2.1.0

FIGURE 12: SAP SYSTEM DETAILS

10.6 Viewing the SAP HANA database status

To display a list of all available SAP HANA databases and their details, proceed as follows:

1. Log in to the Trento Web console.
2. Click the *HANA databases* entry in the left sidebar to show a summary of the state for all SAP HANA databases (see [Figure 13, "HANA databases"](#)).

✓

Passing

3

⚠

Warning

0

❗

Critical

0

Filter Health...

↕

Filter SID...

↕

Filter Tags...

↕

HEALTH	SID	SUMMARY	TAGS
✓	HDD	2 instances: 0 critical, 0 warning, 2 passing	PRD × S4HANA × Add Tag
✓	HDP	2 instances: 0 critical, 0 warning, 2 passing	Add Tag
✓	HDQ	2 instances: 0 critical, 0 warning, 2 passing	Add Tag

1

FIGURE 13: HANA DATABASES

- To investigate the specific SAP HANA database details, click the respective name in the *SID* column to open the corresponding *HANA Database details* view. If the list is too big, reduce the list by providing filters.

Clicking on one of the *SIDs* opens the *HANA Databases* detail view. This view provides the following:


HANA DATABASE DETAILS VIEW

- The *Name* and *Type* of this SAP System.
- Layout* section: lists all related SAP HANA instances with their corresponding virtual host name, instance number, features (roles), HTTP/HTTPS ports, start priorities, and SAPControl status.
- Hosts* section: lists the hosts where all related instances are running. For each host, it provides the host name, the local IP address(es), the cloud provider (when applicable), the cluster name (when applicable), the system ID, and the Trento Agent version.

Clicking on a host name takes you to the corresponding *Host details view*.

HANA Database Details

Name	Type
HDD	HANA Database



Layout

HOSTNAME	INSTANCE NR	FEATURES	HTTP PORT	HTTPS PORT	START PRIO	STATUS
vmhdbdev02	10	HDB HDB_WORKER	51013	51014	0.3	SAPControl: Green
vmhdbdev01	10	HDB HDB_WORKER	51013	51014	0.3	SAPControl: Green

Hosts



HOSTNAME	IP	PROVIDER	CLUSTER	AGENT VERSION
vmhdbdev02	10.100.1.12	 Azure	hana_cluster_1	2.1.0
vmhdbdev01	10.100.1.11 10.100.1.13	 Azure	hana_cluster_1	2.1.0

FIGURE 14: HANA DATABASE DETAILS

11 Housekeeping

When the heartbeat of an agent fails, an option to clean-up the corresponding host will show up both in the *Hosts* overview and the corresponding *Host details* view.

HEALTH	HOSTNAME	IP	PROVIDER	CLUSTER	SID	AGENT VERSION	TAGS	
🔴	vmnwqas03	10.90.1.23 10.90.1.27	Azure		NWQ	2.1.0	➕ Add Tag	🗑️ Clean up

FIGURE 15: CLEAN UP BUTTON IN HOSTS OVERVIEW

Host Details: vmnwqas03

Agent: not running

Node Exporter: running

Clean up

Name

vmnwqas03

Cluster

Agent version

2.1.0

i

CPU Basic

100%

75%

FIGURE 16: CLEAN UP BUTTON IN HOST DETAILS VIEW

When the user clicks on the cleanup button, a box will open asking for confirmation. If the user grants it, all the components discovered by the agent in that particular host, including the host itself and other components that might depend on the ones running on the host, will be removed from the console. For example, if the user cleans up the host where the primary application server of an SAP System is registered, the entire SAP System will be removed from the console.

Similarly, when a registered application or SAP HANA instance is no longer discovered, an option to clean it up will show up both in the corresponding overview and the corresponding details view.

HEALTH	INSTANCE NR	FEATURES	CLUSTER	HOST
✓	00	ABAP GATEWAY ICMAN IGS	not available	vmnwqas04
✓	00	MESSAGESERVER ENQUE	netweaver_clust...	vmnwqas01
ⓘ	01	ABAP GATEWAY ICMAN IGS	not available	vmnwqas03 Clean up
✓	10	ENQREP	netweaver_clust...	vmnwqas02

FIGURE 17: CLEAN UP BUTTON SAP SYSTEMS OVERVIEW

HOSTNAME	INSTANCE NR	FEATURES	HTTP PORT	HTTPS PORT	START PRIO	STATUS
sapnwqas1	00	ABAP GATEWAY ICMAN IGS	50013	50014	3	SAPControl: Green
sapnwqas	00	MESSAGESERVER ENQUE	50013	50014	1	SAPControl: Green
ⓘ sapnwqas	01	ABAP GATEWAY ICMAN IGS	50113	50114	3	SAPControl: Green Clean up
sapnwqer	10	ENQREP	51013	51014	0.5	SAPControl: Green

FIGURE 18: CLEAN UP BUTTON IN SAP SYSTEM DETAILS VIEW

When the user clicks the cleanup button, a box will open asking for confirmation. If the user grants it, the instance will be removed from the console along with any possible dependencies. For example, if the user cleans up the ASCS instance of an SAP system, the entire SAP system will be removed from the console.

12 Managing tags

Tags are a means to label specific objects with a purpose, location, owner, or any other property you like. Such objects can be hosts, clusters, databases, or SAP systems. Tags make it easier to distinguish and show all these different objects, making your lists more readable and searchable. You can use any text you like to create your tags except blank spaces and special characters other than + - = . , _ : and @.

The following subsection shows how you can add, remove, and filter objects based on your tags.

12.1 Adding tags to hosts, clusters, databases, and SAP Systems

To add one or more tags to your objects, do the following:

1. Log in to Trento.
2. In the Trento dashboard, go to the overview corresponding to the object for which you want to create tags. For example, the *Hosts* overview.
3. In the *Hosts* overview, search for the host you want to add a tag to.
4. In the *Tags* column, click the *Add Tag* entry.
5. Enter the respective tag and finish with **Enter** .
6. Repeat this procedure to assign as many tags as you want. It can be on the same or on a different host.

After you have finished the above procedure, the changed host contains one or more tags.

The same principle applies to other objects in Trento, be it *Clusters*, *SAP Systems*, or *HANA Databases*.

12.2 Removing tags

If you want to remove existing tags, click the respective part in the dashboard:

1. Log in to Trento.
2. In the Trento dashboard, go to the overview corresponding to the object for which you want to remove tags. For example, the *Hosts* overview.
3. In the *Hosts* overview, search for the host you want to remove a tag from.
4. In the *Tags* column, click the \times icon to remove the tag.
5. Repeat this procedure to remove as many tags as you want. It can be on the same or on a different host.

12.3 Filter by tags

With tags, you can show only those objects you are interested in. For example, if you have created a drbd tag for some hosts, you can show only those hosts that have this drbd tag assigned.

1. In the Trento dashboard, go to the respective overview you are interested in.
2. Make sure you have already assigned some tags. If not, refer to [Section 12.1, “Adding tags to hosts, clusters, databases, and SAP Systems”](#).
3. In the second row, click the last drop-down list with the name *Filter tags*. The drop-down list opens and shows you all of your defined tags.
4. Select the tag you want to filter from the drop-down list. It is possible to select more than one tag.

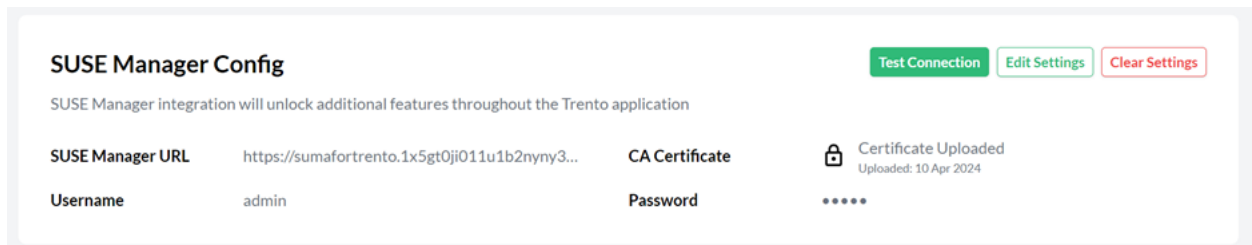
After you have finished the above procedure, Trento shows all respective hosts that contain one or more of your selected tags.

To remove the filter, click the × icon from the same drop-down list.

13 Integration with SUSE Manager

Trento integrates with SUSE Manager to provide the SAP administrator with information about relevant patches and upgradable packages for any host that is registered with both applications.

The user must enter the connection settings for SUSE Manager in the *Settings* view:



The screenshot shows the 'SUSE Manager Config' settings page. At the top, there's a title 'SUSE Manager Config' and three buttons: 'Test Connection' (green), 'Edit Settings' (green), and 'Clear Settings' (red). Below the title, a subtitle states: 'SUSE Manager integration will unlock additional features throughout the Trento application'. The main settings area is divided into two columns. The left column contains 'SUSE Manager URL' with a truncated URL 'https://sumafortrento.1x5gt0ji011u1b2ny3...' and 'Username' with the value 'admin'. The right column contains 'CA Certificate' and 'Password' (masked with dots). To the right of the 'CA Certificate' field, there's a lock icon and a status message: 'Certificate Uploaded' with 'Uploaded: 10 Apr 2024'.

FIGURE 19: SUSE MANAGER SETTINGS

Once the SUSE Manager settings are configured, the SAP Basis administrator can test the connection by clicking the *Test* button. If the connection is successful, the *Host Details* view of each host managed by SUSE Manager displays a summary of available patches and upgradable packages:

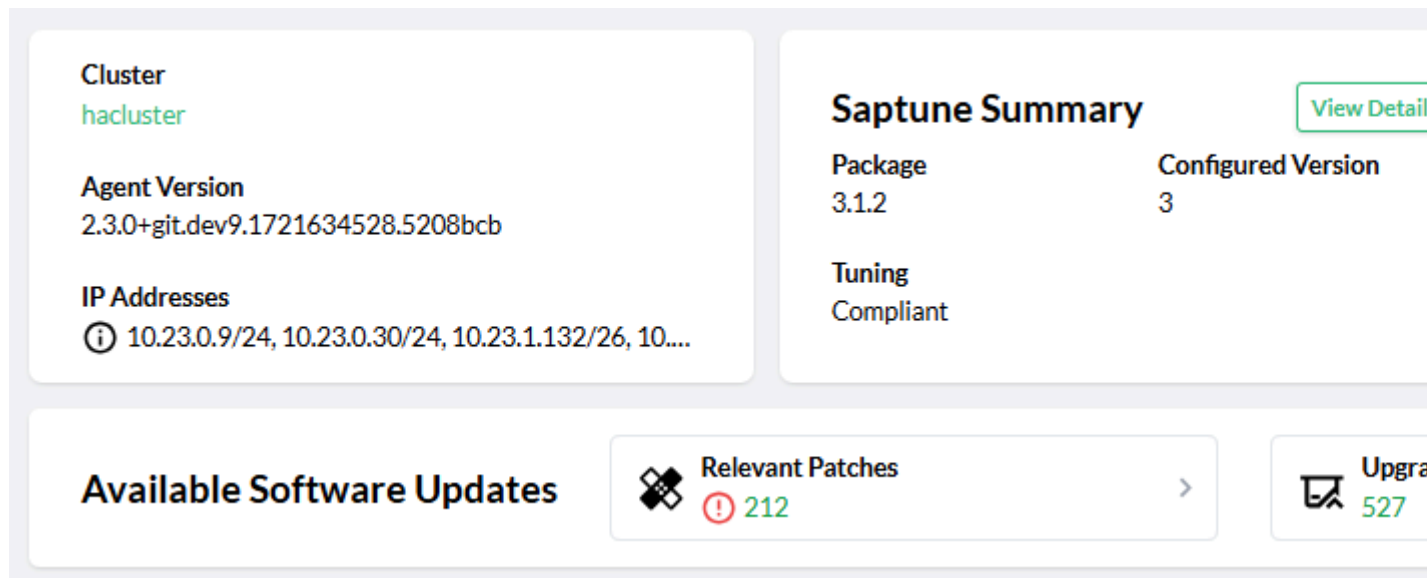












FIGURE 20: AVAILABLE SOFTWARE UPDATES IN THE HOST DETAILS VIEW

Click on *Relevant Patches* to open the *Relevant Patches* overview containing a list of patches available for that particular host:

Relevant Patches: hana-s1-db1

all

Search by Sy

TYPE	ADVISORY	SYNOPSIS
	SUSE-15-SP5-2024-3747	important: Security update for protobuf
	SUSE-15-SP5-2024-3721	Recommended update for libblockdev
	SUSE-15-SP5-2024-3718	Recommended update for libzypp
	SUSE-15-SP5-2024-3713	Recommended update for supportutils-plugin-ha-sap
	SUSE-15-SP5-2024-3659	Recommended update for gcc14
	SUSE-15-SP5-2024-3614	critical: Security update for MozillaFirefox
	SUSE-15-SP5-2024-3606	Recommended update for icewm-theme-branding
	SUSE-15-SP5-2024-3605	Recommended update for grub2
	SUSE-15-SP5-2024-3597	Recommended update for bash
	SUSE-15-SP5-2024-3593	Recommended update for rsyslog

Results per page

10

Showing 1–10 of 212

FIGURE 21: AVAILABLE PATCHES OVERVIEW

Click *Upgradable Packages* to open the *Upgradable Packages* overview containing a list of packages that can be upgraded on that particular host:

Upgradable packages: hana-s1-db1	
INSTALLED PACKAGES	LATEST PACKAGE ↑
aaa_base-84.87+git20180409.04c9dae-150300.10.12.1.x86_64	aaa_base-84.87+git20180409.04c9dae-150300.10.20.1.x86_64
aaa_base-extras-84.87+git20180409.04c9dae-150300.10.12.1.x86_64	aaa_base-extras-84.87+git20180409.04c9dae-150300.10.20.1.x86_64
autofs-5.1.3-150000.7.14.1.x86_64	autofs-5.1.3-150000.7.20.1.x86_64
azure-cli-2.17.1-150100.6.14.2.noarch	azure-cli-2.58.0-150400.14.9.1.noarch
azure-cli-core-2.17.1-150100.6.18.1.noarch	azure-cli-core-2.58.0-150400.14.3.2.noarch
azure-cli-nspkg-3.0.4-6.4.1.noarch	azure-cli-nspkg-3.0.4-150400.13.3.1.noarch
azure-cli-telemetry-1.0.6-3.4.1.noarch	azure-cli-telemetry-1.1.0-150400.10.3.1.noarch
bash-4.4-150400.25.22.x86_64	bash-4.4-150400.27.3.2.x86_64
bash-sh-4.4-150400.25.22.x86_64	bash-sh-4.4-150400.27.3.2.x86_64
bind-utils-9.16.48-150500.8.18.1.x86_64	bind-utils-9.16.50-150500.8.21.1.x86_64
Results per page 10 Showing 1–10 of 527	

FIGURE 22: *UPGRADABLE PACKAGES OVERVIEW*

Advisory Details: SUSE-15-SP5-2024-3747

Synopsis

important: Security update for protobuf

Issued 22 Oct 2024

Status stable

Updated 22 Oct 2024

Reboot Required No

Affects Package Maintenance Stack No

Description

This update for protobuf fixes the following issues: - CVE-2024-7254: Fixed stack overflow vulnerability in Protocol Buffer

Fixes

VUL-0: CVE-2024-7254: protobuf: StackOverflow vulnerability in Protocol Buffers

CVEs

CVE-2024-7254

Affected Packages

libprotobuf-lite25_1_0-25.1-150500.12.5.1-x86_64
libprotoc25_1_0-25.1-150500.12.5.1-x86_64
libprotobuf25_1_0-25.1-150500.12.5.1-x86_64
python311-protobuf-4.25.1-150500.12.5.1-x86_64

Affected Systems

hana-s1-db1
hana-s1-db2
hana-s2-db1
hana-s2-db2
vmhsuangi01

FIGURE 23: ADVISORY DETAILS VIEW

There are three types of patches or advisories: security advisories, bug fixes and feature enhancements. Security advisories are regarded as critical. If one is available, the health of the host will be set to critical. If there are available patches but none of them is a security one, the health of the host will switch to warning. When a host cannot be found in SUSE Manager or there is a problem retrieving the data for it, its health is set as unknown.

At any time, the user can clear the SUSE Manager settings from the Settings view. As a result, all information about available software updates disappears from the console and the status of the hosts is adjusted accordingly.

14 Rotating API keys

The communication from the Trento Agent to the Trento Server is secured by a API key which must be provided in the agent configuration file.

By default, the API key does not have an expiration date. To increase the overall security of the setup, meet internal security requirements, or both, the user can set up a custom expiration date. To do that, go to the Settings view and click the *Generate Key* button in the API Key section:

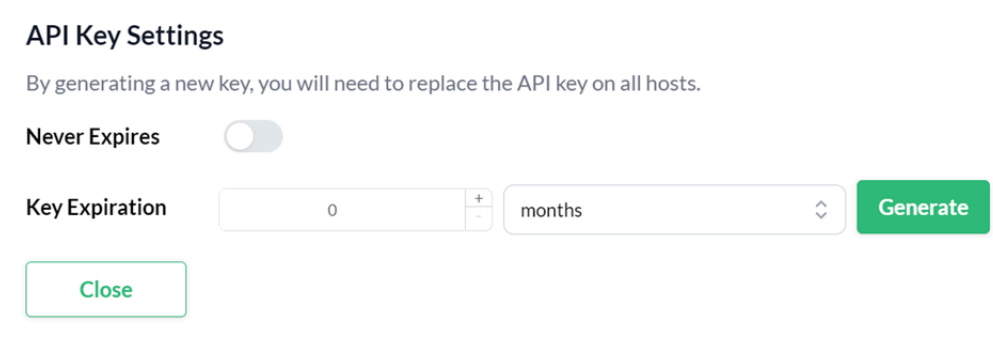


FIGURE 24: CHECKS CATALOG

Whenever a new key is generated, the configuration of all the reporting agents must be updated accordingly.

15 Updating Trento Server

The procedure to update Trento Server depends on the chosen deployment option: Kubernetes, systemd or containerized.

Consider the following when performing an update:

- Before updating Trento Server ensure that all the Trento Agents in the environment are supported by the target version. See section [Section 22, “Compatibility matrix between Trento Server and Trento Agents”](#) for details.
- When updating Trento to version 2.4 or higher, the admin password may need to be adjusted to follow the rules described in the User Managmeent section .

In a Kubernetes deployment, you can use Helm to update Trento Server:

```
helm upgrade \  
--install trento-server oci://registry.suse.com/trento/trento-server \  
--set trento-web.trentoWeb0origin=TRENTO_SERVER_HOSTNAME \  
--set trento-web.adminUser.password=ADMIN_PASSWORD
```

If you have configured options, such as email alerting, the Helm command must be adjusted accordingly. Additional considerations in this scenario:

- Remember to set the helm experimental flag if you are using a version of Helm lower than 3.8.0.
- When updating Trento to version 2.0.0 or higher, an additional flag must be set in the Helm command:

```
helm upgrade \  
--install trento-server oci://registry.suse.com/trento/trento-server \  
--set trento-web.trentoWeb0origin=TRENTO_SERVER_HOSTNAME \  
--set trento-web.adminUser.password=ADMIN_PASSWORD \  
--set rabbitmq.auth.erlangCookie=$(openssl rand -hex 16)
```

- When updating Trento to version 2.3 or higher, a new API key is generated and the configuration of all registered Trento Agents must be updated accordingly.

In a system deployment, you can use zypper to update Trento Server:

```
zypper refresh  
zypper update trento-web  
zypper update trento-wanda  
systemctl restart trento-web  
systemctl restart trento-wanda
```


In a containerized deployment, you can use the same docker commands as for the installation. Keep in mind that the volume for the Trento checks already exists, so there is no need to create it again. Also, the web and wanda containers must be stopped and removed before they can be redeployed with the latest version. In addition, make sure to include in the docker commands any other options that you have enabled after the original installation.

16 Updating a Trento Agent

To update the Trento Agent, do the following:

1. Log in to the Trento Agent host.

2. Stop the Trento Agent:

```
> sudo systemctl stop trento-agent
```

3. Install the new package:

```
> sudo zypper ref
> sudo zypper install trento-agent
```

4. Copy file `/etc/trento/agent.yaml.rpm.save` onto `/etc/trento/agent.yaml` and ensure that entries `facts-service-url`, `server-url`, and `api-key` are maintained properly in the latter.

5. Start the Trento Agent:

```
> sudo systemctl start trento-agent
```

6. Check the status of the Trento Agent:

```
> sudo systemctl status trento-agent
● trento-agent.service - Trento Agent service
   Loaded: loaded (/usr/lib/systemd/system/trento-agent.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2021-11-24 17:37:46 UTC; 4s ago
     Main PID: 22055 (trento)
        Tasks: 10
      CGroup: /system.slice/trento-agent.service
              └─22055 /usr/bin/trento agent start --consul-config-dir=/srv/consul/consul.d
```

```
└─22220 /usr/bin/ruby.ruby2.5 /usr/sbin/SUSEConnect -s
```

```
[...]
```

7. Check the version in the *Hosts* overview of the Trento Web console (URL http://TRENTO_SERVER_HOSTNAME).
8. Repeat this procedure in all Trento Agent hosts.

17 Updating Trento Checks

Configuration checks are an integral part of the checks engine, but they are delivered separately from it. This allows customers to update the checks catalog in their setup whenever updates to existing checks and new checks are released, without waiting for a new version release cycle.

The procedure of updating the configuration checks depends on the Trento Server deployment type: Kubernetes, systemd or containerized.

In a k8s; deployment, checks are delivered as a container image and you can use Helm with the following options to pull the latest one available into your setup:

```
helm ... \  
--set trento-wanda.checks.image.tag=latest \  
--set trento-wanda.checks.image.repository=registry.suse.com/trento/trento-checks \  
--set trento-wanda.checks.image.pullPolicy=Always \  
...
```

In a systemd deployment, checks are delivered as an RPM package, and you can use zypper to update your checks catalog to the latest version:

```
> sudo zypper ref  
> sudo zypper update trento-checks
```

In a containerized deployment, checks are delivered as a container image, and you can use Docker to pull the latest version into the trento-checks volume that was created during the installation process:

```
> docker run \  
-v trento-checks:/usr/share/trento/checks \  
registry.suse.com/trento/trento-checks:latest
```

18 Uninstalling Trento Server

The procedure to uninstall the Trento Server depends on the deployment type: Kubernetes, systemd or containerized. The section covers Kubernetes deployments.

If Trento Server was deployed manually, then you need to uninstall it manually. If Trento Server was deployed using the Helm chart, you can also use Helm to uninstall it as follows:

```
helm uninstall trento-server
```

19 Uninstalling a Trento Agent

To uninstall a Trento Agent, perform the following steps:

1. Log in to the Trento Agent host.

2. Stop the Trento Agent:

```
> sudo systemctl stop trento-agent
```

3. Remove the package:

```
> sudo zypper remove trento-agent
```

20 Reporting a Problem

Any SUSE customer, with a registered SUSE Linux Enterprise Server for SAP Applications 15 (SP3 or higher) distribution, experiencing an issue with Trento, can report the problem either directly in the SUSE Customer Center or through the corresponding vendor, depending on their licensing model. Problems must be reported under SUSE Linux Enterprise Server for SAP Applications 15 and component trento.

When opening a support case for Trento, provide the chosen deployment option for Trento Server: Kubernetes, systemd or containerized.

In case of a Kubernetes deployment, provide the output of the Trento support plugin and the scenario dump, as explained in section [Section 21, “Problem Analysis”](#).

In case of a systemd deployment, provide the status and the journal of the trento-web and trento-wanda services.

In case of a containerized deployment, provide the logs of the `trento-web` and `trento-wanda` containers. Use `docker ps` to retrieve the IDs of both containers, then `docker logs CONTAINER_ID` to retrieve the corresponding logs.

For issues with a particular Trento Agent, or a component discovered by a particular Trento Agent, also provide the following:

- The status of the Trento Agent.
- The journal of the Trento Agent.
- The output of the command `supportconfig` in the Trento Agent host. See <https://documentation.suse.com/sles/html/SLES-all/cha-adm-support.html#sec-admsupport-cli> for information on how to run this command from command line.

21 Problem Analysis

This section covers some common problems and how to analyze them.

21.1 Trento Server

There are two tools you can use to analyze issues with Trento Server in a Kubernetes scenario. They both help us collect information/data that might be useful when troubleshooting/analyzing the issue.

21.1.1 Trento Support Plugin

The Trento support plugin automates the collection of logs and relevant runtime information on the server side. Using the plugin requires a host with the following setup:

- The packages `jq` and `python3-yq` are installed.
- Helm is installed.
- The command `kubectl` is installed and connected to the K8s cluster where Trento Server is running.

To use it, proceed as follows:

1. Download the Trento support plugin script:

```
# wget https://raw.githubusercontent.com/trento-project/support/refs/heads/main/trento-support.sh
```



Note: Package available in SUSE Linux Enterprise Server for SAP Applications 15 SP3 and higher

Customers of SUSE Linux Enterprise Server for SAP Applications 15 SP3 and higher can install the `trento-supportconfig-plugin` package directly from the official SLES for SAP 15 repos using Zypper. The Containers Module is required for installation.

2. Make the script executable:

```
# chmod +x trento-support.sh
```

3. Ensure `kubect1` is connected to the K8s cluster where Trento Server is running and execute the script:

```
# ./trento-support.sh --output file-tgz --collect all
```

4. Send the generated archive file to support for analysis.

The Trento support plugin admits the following options:

-o, --output

Output type. Options: stdout|file|file-tgz

-c, --collect

Collection options: configuration|base|kubernetes|all

-r, --release-name

Release name to use for the chart installation. "trento-server" by default.

-n, --namespace

Kubernetes namespace used when installing the chart. "default" by default.

--help

Shows help messages

21.1.2 Scenario dump

The scenario dump is a dump of the Trento database. It helps the Trento team to recreate the scenario to test it. Using the script that generates the dump requires a host with the following setup:

- The command **kubectl** is installed and connected to the K8s cluster where Trento Server is running.

To generate the dump, proceed as follows:

1. Download the latest version of the dump script:

```
> wget https://raw.githubusercontent.com/trento-project/web/main/hack/dump_scenario_from_k8.sh
```

2. Make the script executable:

```
> chmod +x dump_scenario_from_k8.sh
```

3. Ensure **kubectl** is connected to the K8s cluster where Trento Server is running and execute the script as follows:

```
> ./dump_scenario_from_k8.sh -n SCENARIO_NAME -p PATH
```

4. Go to `/scenarios/`, package all the JSON files and send them to support for analysis.

21.1.3 Pods descriptions and logs

In addition to the output of the Trento Support Plugin and the Dump Scenario script, the descriptions and logs of the pods of the Trento Server can also be useful for analysis and troubleshooting purposes. These descriptions and logs can be easily retrieved with the **kubectl** command, for which you need a host where command **kubectl** is installed and connected to the K8s cluster where Trento Server is running.

1. List the pods running in Kubernetes cluster and their status. Trento Server currently consists of the following six pods:

```
> kubectl get pods
trento-server-prometheus-server-*
trento-server-postgresql-0
```

```
trento-server-web-*  
trento-server-wanda-*  
trento-server-rabbitmq-0
```

2. Retrieve the description of one particular pod as follows:

```
> kubectl describe pod POD_NAME
```

3. Retrieve the log of one particular pod as follows:

```
> kubectl logs POD_NAME
```

4. Monitor the log of one particular pod as follows:

```
> kubectl logs POD_NAME --follow
```

21.2 Trento Agent

The first source to analyze issues with the Trento Agent is the Trento Agent status, which can be retrieved as follows:

```
> sudo systemctl status trento-agent
```

If further analysis is required, it is convenient to activate debug log level for the agent. A detailed log can be then retrieved from the journal:

1. Add parameter `log-level` with value `debug` to the `/etc/trento/agent.yaml` configuration file.

```
> sudo vi /etc/trento/agent.yaml
```

2. Add the following entry:

```
log-level: debug
```

3. Restart the agent:

```
> sudo systemctl restart trento-agent
```

4. Retrieve the log from the journal:

```
> sudo journalctl -u trento-agent
```

22 Compatibility matrix between Trento Server and Trento Agents


TABLE 1: COMPATIBILITY MATRIX BETWEEN TRENTO SERVER AND TRENTO AGENTS

Trento Agent Compatibility matrix		Trento Server versions							
		1.0	1.1	1.2	2.0	2.1	2.2	2.3	2.4
Trento Agent ver- sions	1.0	✓							
	1.1	✓	✓						
	1.2	✓	✓	✓					
	2.0				✓				
	2.1				✓	✓			
	2.2				✓	✓	✓		
	2.3				✓	✓	✓	✓	
	2.4				✓	✓	✓	✓	✓

23 Highlights of Trento versions

The following list shows the most important user-facing features in the different versions of Trento. For a more detailed information about the changes included in each new version, visit the GitHub project at <https://github.com/trento-project>. Particularly:

- For changes in the Trento Helm Chart, visit <https://github.com/trento-project/helm-charts/releases>.
- For changes in the Trento Server Web component, visit <https://github.com/trento-project/web/releases>.
- For changes in the Trento Server old checks engine component (runner), visit <https://github.com/trento-project/runner/releases>.
- For changes in the Trento Server new checks engine component (wanda), visit <https://github.com/trento-project/wanda/releases>.

- For changes in the Trento checks, visit <https://github.com/trento-project/checks/releases> .
- For changes in the Trento Agent, visit <https://github.com/trento-project/agent/releases> .

Version 2.4.0 (released on 2024/12/06)

Consisting of Helm chart 2.4.0, Web component 2.4.0, checks engine (wanda) 1.4.0, configuration checks 1.0.0 and agent 2.4.0.

- New permission-based user management system with optional MFA.
- SSO and third-party IDP integration via OIDC, OAuth2 and SAML protocols.
- Contextual, detailed overview of available patches and upgradable packages in the SUSE Manager integration.
- Inception of the Activity Log, allowing users to browse a single centralized event log for their entire SAP landscape.
- Trento Checks delivery is now decoupled from the core platform, and the Checks themselves have been relicensed from Apache 2.0 to GPL 3.0
- Enhanced discovery capabilities including clusters using angi architecture and SAP systems running on JAVA stacks.
- New HANA scale-out cluster configuration checks.
- More ASCS/ERS cluster configuration checks.

Version 2.3.2 (released on 2024/09/01)

Consisting of Helm chart 2.3.2, Web component 2.3.2, checks engine (wanda) 1.3.0 and agent 2.3.0.

- Fix for bug in the web component causing it to crash upon certain events after update from an earlier version.

Version 2.3.1 (released on 2024/06/18)

Consisting of Helm chart 2.3.1, Web component 2.3.1, checks engine (wanda) 1.3.0 and agent 2.3.0.

- non-K8s installation of Trento Server.
- Enhanced discovery capabilities, including improved support HANA multi-tenant architecture, HANA scale out cluster discovery, discovery of clusters using diskless SBD and maintenance awareness.

- Information about available patches and upgradable packages for each registered hosts via integration with SUSE manager.
- Rotating API key.
- Saptune configuration checks.
- Simpler, more secure architecture without Grafana.
- Revamped metric visualization.
- Enhanced Checks Catalog view with dynamic filtering capabilities.

Version 2.2.0 (released on 2023/12/04)

Consisting of Helm chart 2.2.0, Web component 2.2.0, checks engine (wanda) 1.2.0 and agent 2.2.0.

- saptune Web integration.
- Intance clean-up capabilities.
- Ability to run host-level configuration checks.

Version 2.1.0 (released on 2023/08/02)

Consisting of Helm chart 2.1.0, Web component 2.1.0, checks engine (wanda) 1.1.0 and agent 2.1.0.

- ASCS/ERS cluster discovery, from single-sid, two-node scenarios to multi-sid, multi-node setups. The discovery covers both versions of the enqueue server, ENSA1 and ENSA2, and both scenarios with resource managed instance filesystems and simple mount setups.
- Host clean-up capabilities, allowing users to get rid of hosts that are no longer part of their SAP environment.
- New checks results view that leverages the potential of the new checks engine (wanda) and provides the user with insightful information about the check execution.

Version 2.0.0 (released on 2023/04/26)

Consisting of Helm chart 2.0.0, Web component 2.0.0, checks engine (wanda) 1.0.0 and agent 2.0.0.

- A brand-new safer, faster, lighter SSH-less configuration checks engine (wanda) which not only opens the door to configuration checks for other HA scenarios (ASCS/ER, HANA scale-up cost optimized, SAP HANA scale-out) and other targets in the environment (hosts, SAP HANA databases, SAP NetWeaver instances), but also will allow customization of existing checks and addition of custom checks.
- Addition of VMware to the list of known platforms.
- Versioned external APIs for both the Web and the checks engine (wanda) components. The full list of available APIs can be found at <https://www.trento-project.io/web/swaggerui/> and <https://www.trento-project.io/wanda/swaggerui/> respectively.

Version 1.2.0 (released on 2022/11/04)

Consisting of Helm chart 1.2.0, Web component 1.2.0, checks engine (runner) 1.1.0 and agent 1.1.0.

- Configuration checks for HANA scale-up performance optimized two-node clusters on on-premise bare metal platforms, including KVM and Nutanix.
- A dynamic dashboard that allows you to determine, at a glance, the overall health status of your SAP environment.

Version 1.1.0 (released on 2022/07/14)

Consisting of Helm chart 1.1.0, Web component 1.1.0, checks engine (runner) 1.0.1 and agent 1.1.0.

- Fix for major bug in the checks engine that prevented the Web component to load properly.

Version 1.0.0 (general availability, released on 2022/04/29)

Consisting of Helm chart 1.0.0, Web component 1.0.0, checks engine (runner) 1.0.0 and agent 1.0.0.

- Clean, simple Web console designed with SAP basis in mind. It reacts in real time to changes happening in the backend environment thanks to the event-driven technology behind it.
- Automated discovery of SAP systems, SAP instances and SAP HANA scale-up two-node clusters
- Configuration checks for SAP HANA scale-up performance optimized two-node clusters on Azure, AWS and GCP.

- Basic integration with Grafana and Prometheus to provide graphic dashboards about CPU and memory utilization in each discovered host.
- Basic alert emails for critical events happening in the monitored environment.

24 More information

- Homepage: <https://www.trento-project.io/> 
- Trento project: <https://github.com/trento-project/web> 