SaltStack Enterprise installation

Summary: This site contains instructions to download and install SaltStack Enterprise and packages for AIX, Solaris, and Windows.

Table of Contents

Prerequisites

SaltStack Enterprise integrates seamlessly with a new or existing Salt installation.

SaltStack Enterprise Server requirements

Set up a server to host SaltStack Enterprise. This server hosts the Enterprise API and Enterprise Console web components, and optionally a Salt Master.

Supported operating systems for SaltStack Enterprise:

The following operating systems are supported to host your SaltStack Enterprise server:

- Red Hat Enterprise Linux 7 (RHEL 7)/CentOS 7 (Cent7)
- SUSE Linux Enterprise Server 12 SP3 (SLES)/openSUSE Leap 42.3

  The SaltStack Enterprise SUSE packages will come with the 5.5 patch later this year.

> This list is for the SaltStack Enterprise server, not for the Salt Masters in your environment. See Supported Salt Versions for a list of supported Salt Master operating systems.

Supported Python versions for SaltStack Enterprise:

Python 3.5.3 or later is required on your SaltStack Enterprise server. Python 3.x packages are provided with the SaltStack Enterprise installation files.

SaltStack Enterprise cannot run on Salt Masters running Python 3. Support for masters running Python 3 is planned for the 1Q2019 release. For best results, use Python 2.7 on your Salt Masters.

SaltStack Enterprise Licensing

SaltStack Enterprise requires a license file to track minion usage and duration of contract. **The SaltStack Enterprise download contains a 14-day trial license. After 14 days the Enterprise API service no longer starts.**

Customers receive a license file with the Welcome letter from SaltStack Support. If you are a current customer and have not received a license file, or if you encounter any issues with the licensing process, please contact SaltStack support.

Before 14 days, your license file must be placed on your SaltStack Enterprise server at `/etc/raas/raas.license` for continued functionality.

Hardware recommendations for single-node installation

Requirements for installing the Salt Master, SaltStack Enterprise, and PostgreSQL on the same host:

Up to 500 (minimum)	Up to 500 (recommended)	500 to 1000 (minimum)	500 to 1000 (recommended)
2 CPU cores*	4 CPU cores	6 CPU cores	8 CPU cores
2 GB RAM	8 GB RAM	8 GB RAM	8 GB RAM
At least 20 GB free disk space**	At least 20 GB free disk space**	At least 20 GB free disk space**	At least 20 GB free disk space**

* If using the scheduler you should have at least 4 GB of RAM.

** Used for minion return data. Increase according to your needs for data retention.

Hardware recommendations for multi-node installation

Multi-node is recommended for environments with more than 1000 minions.

Requirements for installing the Salt Master, SaltStack Enterprise, and PostgreSQL on separate hosts:

	1000 to 2500 minions	2500 to 5000 minions	Greater than 5000 minions

	1000 to 2500 minions	2500 to 5000 minions	Greater than 5000 minions
Salt Master node	4 CPU cores 8 GB RAM	8 CPU cores 16 GB RAM	Consider multiple masters
SaltStack Enterprise node	4 CPU cores 8 GB RAM	8 CPU cores 16 GB RAM	Create an additional SaltStack Enterprise node per 5000 minions, hosted behind your preferred load-balancing solution
PostgreSQL node	4 CPU cores 8 GB RAM At least 40 GB free disk space*	8 CPU cores 16 GB RAM At least 80 GB free disk space*	Increase PostgreSQL CPU cores and RAM, as indicated by performance
Redis node	4 CPU cores 8 GB RAM	8 CPU cores 16 GB RAM	Increase Redis CPU cores and RAM, as indicated by performance

* Used for minion return data. Increase according to your needs for data retention.

Database requirements

> SaltStack Enterprise requires a PostgreSQL 9.5 or 9.6 database. PostgreSQL 9.6 is recommended. PostgreSQL 10 is not supported.

Important notes regarding the SaltStack Enterprise database:

1. The database is supported on PostgreSQL only.
2. The SaltStack Enterprise installer can install and configure PostgreSQL on your SaltStack Enterprise server, however, you are responsible for ongoing maintenance, backups, and other administrative tasks for this database. See the PostgreSQL documentation for details on PostgreSQL database maintenance and administration.
3. A PostgreSQL tuning guide can be found here

Supported Salt versions

SaltStack Enterprise can be used with the currently supported open source versions of Salt. See SaltStack Platform Support for a list.

Supported web browsers for SaltStack Enterprise Web Console

The latest versions of Google Chrome and Mozilla Firefox are supported.

Changes to your Salt environment

Installing SaltStack Enterprise makes the following changes to your Salt environment:

- Enterprise API backend services (file system, pillar store, and so on) take precedence over any other existing backends defined in your environment. You can continue to use all supported backend services, just be aware that files that exist in Enterprise Console will take precedence if they also exist in other file or pillar backends (See the Configuration topic in the SaltStack Enterprise help to learn how to change this behavior).
- Enterprise API replaces the Salt Master syndic component to provide Salt Minion aggregation and scale. Salt Syndic Masters are not compatible with the SaltStack Enterprise architecture. Instead, each root Salt Master should connect directly to the Enterprise API.

Existing Salt States, configuration settings, and Salt Minion connections are unchanged. No changes are required on the Salt Minion to use SaltStack Enterprise.

Tuning Processes on your SaltStack Enterprise Server

When the SaltStack Enterprise Service (*raas*) starts, it creates two types of processes:

- **Tornado processes** - allows connections from Salt Masters and web browsers
- **Celery processes** - background workers

By default, raas sets the count for each process type to half the number of CPU Cores.

In most cases this is optimal, as the raas host should be dedicated to this task.

If you need to deploy raas on a host that supports additional services, you can override the default behavior by adding the following to your `/etc/raas/raas` configuration file.

```
num_processes: 8
background_workers:
  concurrency: 8
```

Known issues

- If the Postgres DB is not set to use UTF-8, sorting will not be consistent across the application.
- Targets created through the API can display report data in the web console only if created under the compound target type.
- Daemon support was removed this release. The system packages contain an init script that starts SaltStack Enterprise and leaves it running. You can use the system start script to run the service as a daemon.
- `raas -d` should not be used with this release of SaltStack Enterprise, it will result in loop creation of worker processes (too many worker processes will be created).
- Non-scheduled jobs will capture the job execution time in the master time zone. Scheduled jobs will capture job execution time in UTC. To sync these times, the master time zone must be set to UTC or there will be a discrepancy between jobs run at the same time on a schedule vs. from the command line.
- Salt-SSH commands are running longer in SaltStack Enterprise than through the Salt CLI.
- SaltStack Enterprise cannot run if it is not connected to Redis. If the first attempt to connect is unsuccessful, it does not retry automatically.
- Schedules cannot be edited if their parent job name has changed.
- Schedules can be deleted only by the schedule author if the author has `Delete Schedules` permission, or by the `SuperUser`. All other users cannot delete schedules they did not create, even if they have been granted `Delete Schedules` permission.
- It is possible to remove the `root` user from the `SuperUser` group, leaving no users in the group. This is strongly discouraged as it is not possible to grant `SuperUser` access from a non-superuser.
- Scheduled jobs display in the web console only if scheduled within the next 12 weeks.
- Postgres users do not have sufficient privileges to run commands through the firehose, which triggers an error.
- Modifications to sample jobs, files, or default targets are overwritten on re-installation or upgrade. To preserve changes, make sure to save modified jobs and targets with a new name, and save modified files with a new name or environment.
- When viewing job history for orchestrate job run on one master, it shows results on all masters.
- Job return numbers may differ from target numbers based on current key state and grain data.

Install using the SaltStack Enterprise installer

The script installs all necessary dependencies and then applies Salt States to install SaltStack Enterprise. The required versions of PostgreSQL, Redis, PyOpenSSL, and Python Setuptools have been included for your convenience.
This is helpful for installations where servers do not have direct internet access.

RHEL 7 installation

Download the installer files on the SaltStack Enterprise website .

Complete steps below for either a single node or multi-node installation.

- Single node
- Multi-node

Single node

> If your version of RHEL 7 is lower than 7.4, you will need to update your OpenSSL version to 1.0.2k before running the installation script.
>
> If this version is not available to you via a `yum` update, or your server does not have direct internet access, retrieve the following packages from Red Hat or from your preferred public mirror:
>
> - openssl-1.0.2k-12.el7.x86_64.rpm
> - openssl-libs-1.0.2k-12.el7.x86_64.rpm

Use this method if you want to install the Salt Master, SaltStack Enterprise, Redis, and PostgreSQL on the same node. This is appropriate for installations with up to 1,000 minions.

For installations with more than 1,000 minions, please perform a multi-node installation. See Multi-node installation.

1. Expand the tarball.

   ```
   $ tar zxf SaltStack_Enterprise_5.5_Installer.tar.gz
   $ cd sse_installer
   ```

2. Run the command:

   ```
   $ sudo ./setup_single_node.sh
   ```

   This script configures a Salt Master and Salt Minion. It then installs PostgreSQL, Redis, SaltStack Enterprise, and the Salt Master Plugin on the same server.

   > This should be a fresh installation of RHEL. Ideally, Salt should not yet be installed.
   >
   > If both the Salt Master and Salt Minion are installed, the script skips this step and proceeds with the setup of SaltStack Enterprise.
   >
   > If either the Salt Master or the Salt Minion packages are installed, but not both, the script will terminate.
   >
   > This protects the user from accidentally disrupting an existing installation.

3. Confirm that you can log in to SaltStack Enterprise.

   Log in to the web console using your browser (Chrome is recommended).
   The default installation uses `https://` and generates a self-signed certificate.

   The default credentials are as follows:

   - URL: `https://IP_or_DNS_name_of_your_Server`
   - Username: `root`
   - Password: `salt`

> The `setup_single_node.sh` script does not modify firewall rules.

> Please ensure that access is allowed to port 443 in your firewall rules for all appropriate systems (Salt Masters, web-based interface users, remote systems calling the Enterprise API, etc).

Multi-node

Use this method when installing SaltStack Enterprise on a distributed system.
This method is required for installations with more than 1,000 minions, but is perfectly appropriate for smaller installations.

For a multi-node installation, you will need to integrate the pillar and configuration states into your existing environment (these are provided within the installation tarball).

The starting point for this procedure is that you have created the following node types:

- Salt Master
- PostgreSQL
- Redis
- SaltStack Enterprise API (eAPI)

Each of these servers must be a Salt Minion of the Salt Master.

1. On the Salt Master, expand the tarball.

   ```
   $ sudo tar zxf SaltStack_Enterprise_5.5_Installer.tar.gz
   $ sudo cd sse_installer
   ```

2. Copy the pillar and state files into your `pillar_roots` and `file_roots` location.

   For example, in a default Salt Master configuration where the pillar and configuration state file roots are `/srv/pillar` and `/srv/salt`, the commands to copy the related files into their correct locations would be:

   ```
   $ sudo mkdir /srv/salt
   $ sudo cp -r salt/sse /srv/salt/
   $ sudo mkdir /srv/pillar
   $ sudo cp -r pillar/sse.sls /srv/pillar/
   ```

   If this is a brand new installation, make sure to create `/srv/pillar` and `/srv/salt` with the following commands:

   ```
   $ sudo mkdir /srv/pillar
   $ sudo mkdir /srv/salt
   ```

   > This assumes that you do not already have a folder named “sse” for some unrelated purpose under either your pillar or configuration state root.

3. Create or update your existing Pillar “Top” file

   Create or update your `/srv/pillar/top.sls` file with the content from the provided `sse_install/pillar/top.sls` file. Define the list of minion IDs for your PostgreSQL, Redis, eAPI, and Salt Masters.

   For example:

```
{# Pillar Top File #}

{# Define SSE Servers #}

{% load_yaml as sse_servers %}
  - saltpgsql
  - saltredis
  - salteapi
  - saltmaster
{% endload %}

base:

{# Assign Pillar Data to SSE Servers #}
{% for server in sse_servers %}
'{ { server }}':
  - sse
{% endfor %}
```

4. Update `pillar/sse/sse_settings.yaml` with the values appropriate for your environment. These settings will be used by the configuration state files to deploy and manage your SaltStack Enterprise deployment.

    - *Section 1*

      You will need to provide the Minion ID (as opposed to the IP or DNS name) for each server type. Please note that `pg_server` and `redis_server` items are single values. The `eapi_servers` and `salt_masters` items are lists, as these two server types have high-availability deployment options supported by the installation states.

    - *Section 2*

      You will need to specify the `pg_endpoint` for your PostgreSQL server. For this option, be sure to specify the DNS name or IP address for your PostgreSQL server (not the Minion ID). The standard PostgreSQL port is provided, but may be overridden, if desired.

        - This is specified as the `pg_endpoint` as some installations may have configured a separate PostgreSQL server (or cluster) that is not managed by this installation process. If that is the case, you will want to exclude the action to highstate the PostgreSQL server in step 8 of this guide.

        - If you are in a virtualized environment, take care to specify the *internal* address, as opposed to the *public* address.

      You will also specify the username and password for the PostgreSQL user that will be used by the eAPI server(s) to authenticate to PostgreSQL.
      This user will be created for you when you run the configuration states.

    - *Section 3*

      You will need to specify the `redis_endpoint` for your Redis server.
      The standard Redis port is provided, but may be overridden.

      The `redis_username` and `redis_password` are also specified in this section.

    - *Section 4*

      You will next define the configuration settings for your eAPI servers.
      The initial `eapi_username` and `eapi_password` values are `root` and `salt`, respectively.

        - If this is a fresh installation, it is important that you *do not change* these values. During the initial run of these states, the installation process will establish the database with these default credentials then connect through the eAPI service to establish your default Targets and Jobs.

        - After your initial deployment is completed and you have tested your access to the web-based user interface, then you are *strongly advised* to do the following:

            1. Update the `root` user's password via the web-based user interface.
            2. Update `/srv/pillar/sse/sse_settings.yaml` with the new password.
            3. Reapply the highstate on your Salt Master(s).

You will need to specify the `eapi_endpoint` for your SaltStack Enterprise server.
For this option, be sure to specify the DNS name or IP address for your eAPI server (not the Minion ID).

- This is referred to as the `eapi_endpoint`, as some installations host multiple eAPI servers behind a load balancer.
- You may also specify whether or not SSL should be enabled on the eAPI servers and if the SSL certificate should be validated. It is *strongly recommended* to enable SSL. SSL validation is not required by the installer, but is likely a security requirement in environments that host their own certificate authority.
- The `eapi_standalone` option is present to provide direction to the configuration states if Pillar data is being used in a single node deployment. In that event, all IP communication would be directed to the loopback address. Since you are using this guide, you should leave this set to `False`.
- The `eapi_deploy_default_spm` option is present to suppress the deployment of the default Jobs, Targets, or files in the SSE Filesystem provided by SaltStack Enterprise. If are deploying an update to an existing installation and you have modified any of these items, you will likely want to set this to `False`. Otherwise, `True` is recommended.
- The `eapi_failover_master` option is present to support deployments where Salt Masters (and Salt Minions) are operating in "Failover" mode. For Multi-Master configurations, SaltStack strongly recommends use of "Active" Multi-Master configurations.
- The `eapi_key` option is present to allow the user to define the encryption key that SaltStack Enterprise uses to manage encrypted data in the PostgreSQL database. This key should be unique for each installation.
  A default is provided, but a custom key can be generated by running the following command:

  ```
  openssl rand -hex 32
  ```

- *Section 5*

  The `customer_id` value uniquely identifies a SaltStack deployment.
  Primarily, it becomes the suffix of the schema name of the `raas_*` database in PostgreSQL. A default is provided, but a custom key can be generated by running the command:

  ```
  cat /proc/sys/kernel/random/uuid
  ```

  The `cluster_id` value defines the ID for a set of Salt Masters, when configured in either "Active" or "Failover" Multi-Master mode. This prevents Salt Minions that are reporting to multiple Masters from being reported multiple times in the Targets view within the SaltStack Enterprise.

5. Create or Update your existing Configuration State "Top" file.

   Create or update your `/srv/salt/top.sls` file with the content from the provided `sse_install/salt/top.sls` file.

   The syntax within will leverage the Pillar data provided in *Section 1* to provide the Minion IDs of the nodes that will require the SSE Pillar data.

   For example:

```
base:

  {# Target SSE Servers, according to Pillar data #}

  # SSE PostgreSQL Server
  'I@sse_pg_server:':
    - sse.eapi_database

  # SSE Redis Server
  'I@sse_redis_server:':
    - sse.eapi_cache

  # SSE eAPI Servers
  'I@sse_eapi_servers:':
    - sse.eapi_service

  # SSE Salt Masters
  'I@sse_salt_masters:':
    - sse.eapi_plugin
```

6. Sync Grains

   For Pillar data to be properly generated, we must confirm that the Salt Master has all grain data from each of the Minions that will provide a part of the SaltStack Enterprise functionality.

   ```
   $ sudo salt -L '[LIST_OF_SSE_RELATED_NODES]' saltutil.refresh_grains
   ```

7. Refresh and Confirm Pillar Data

   Prior to running the SaltStack Enterprise deployment via hightate, confirm that each of the SaltStack Enterprise related nodes has received the Pillar data defined in the `sse_settings.yaml` file and that it appears as expected.

   ```
   $ sudo salt -L '[LIST_OF_SSE_RELATED_NODES]' saltutil.refresh_pillar
   ```

   If your Pillar data appears to be correct, proceed with the next step.

8. Apply the highstate to the following servers:

   - PostgreSQL Server
   - Redis Server
   - SaltStack Enterprise Server(s)
   - Salt Master(s)

```
$ sudo salt <MINION_ID_OF_RELATED_SERVER> state.highstate
```

9. Confirm that you can log in to SaltStack Enterprise

   Log in to the web-based interface using your browser (Chrome is recommended).
   The default installation uses `https://` and generates a self-signed certificate.

   The default credentials are as follows:

   - URL: `https://IP_or_DNS_name_of_your_Server`
   - Username: `root`
   - Password: `salt`

The SaltStack Enterprise Installer does not modify firewall rules.

Please ensure that firewall access is allowed on the following ports from the following nodes:

- PostgreSQL is accessible by (5432 by default)
    - eAPI servers
- Redis is accessible by (6739 by default)
    - eAPI Servers
- eAPI endpoint is accessible by (443 by default)
    - Salt Masters
    - Web-based interface users
    - Remote systems calling the Enterprise API
- Salt Masters are accessible by (4505/4506 by default)
    - All Salt Minions configured to use the related Salt Master

Install SaltStack Enterprise manually

These instructions walk you through installing Enterprise API without using the installation states. These instructions are intended for advanced users who need granular control over the installation process, and who are familiar with PostgreSQL and Redis database configuration.

The steps below are confirmed for a standalone deployment of SaltStack Enterprise (where all related services reside on a single host). As an advanced user, you will likely adapt these instructions to your deployment. If you are not an advanced user, consider using the deployment states provided by installer. See *Use the installer*.

> SaltStack Enterprise requires a PostgreSQL 9.5 or 9.6 database. PostgreSQL 9.6 is recommended. PostgreSQL 10 is not supported.
>
> The SaltStack Enterprise SUSE packages will come with the 5.5 patch later this year.

Download the packages for your environment

Download the packages on the SaltStack Enterprise website.

Red Hat Enterprise Linux 7/CentOS 7

Step 1: PostgreSQL database installation and configuration

1. Install PostgreSQL.

```
# run one of these commands based on your OS
Red Hat
$ sudo wget https://download.postgresql.org/pub/repos/yum/9.6/redhat/rhel-7-x86_64/pgdg-redhat96-9.6-3.noarch.rpm

CentOS
$ sudo wget https://download.postgresql.org/pub/repos/yum/9.6/redhat/rhel-7-x86_64/pgdg-centos96-9.6-3.noarch.rpm

# run all of these commands
$ sudo yum install pgdg-*.noarch.rpm
$ sudo yum update
$ sudo yum install postgresql96-server
$ sudo yum install postgresql96-contrib
$ sudo /usr/pgsql-9.6/bin/postgresql96-setup initdb
```

2. Update the `pg_hba.conf` file as needed to enable connections from your SaltStack Enterprise server. Optionally, enable ssl.

3. Start PostgreSQL and create a database account for Enterprise API, for example:

```
$ sudo systemctl enable postgresql-9.6
$ sudo systemctl start postgresql-9.6
$ sudo su - postgres -c 'createuser -s -P salt_eapi'
# This account has Superuser privileges so that
# various extensions my be installed.
# After initial deployment the Superuser privilege
# may be removed.
```

Step 2: Redis installation and configuration

1. Download the `Redis` and `jemalloc` installation packages for RH/CentOS that are provided in the download section.

```
$ sudo yum install redis40u-4.0.11-1.ius.el7.x86_64.rpm jemalloc-3.6.0-1.el7.x86_64.rpm
```

2. *Optional*: Update configuration

   If you are setting up Redis on a host that is separate from the SaltStack Enterprise Server, you will need to configure Redis to accept remote connections and to limit access using a password. To do this, update the `/etc/redis.conf` file, specifying the bind parameter and setting the password that your SaltStack Enterprise servers should use to authenticate.

```
bind 0.0.0.0
requirepass
```

3. Start the Redis service

```
$ sudo systemctl enable redis
$ sudo systemctl start redis
```

Step 3: SaltStack Enterprise installation and configuration

1. Download the `Python3.5` and `libpython3.5` installation packages for RH/CentOS that are provided in the download section.

```
$ sudo yum install python35u-libs-3.5.4-1.*.rpm python35u-3.5.4-1.*.rpm
```

2. Download and install the Red Hat/CentOS SaltStack Enterprise RPM.

```
$ sudo yum install raas-5.5-0.el7.x86_64.rpm
```

3. Update RaaS Configuration File

   Open `/etc/raas/raas` in a text editor and update the `sql` configuration to provide the host, port, and the username and password created in the previous section. If you plan to use SSL, set `ssl` to `True.`

```
sql:
  dialect: postgresql
  username: salt_eapi
  password: abc123
  host: localhost
  port: 5432
  driver: psycopg2
  ssl: false
```

4. Start the Enterprise API service.

```
$ sudo systemctl enable raas
$ sudo systemctl start raas
```

5. Confirm that you can connect to the web console in a web browser.

   - **url:** `http://your_raas_server/`
   - **Username:** `root`

- Password: `salt`

Enable SSL on Red Hat Enterprise Linux 7/CentOS 7 (optional)

1. Install pyOpenSSL.

```
Red Hat/CentOS
$ sudo yum install pyOpenSSL
```

2. Enable SSL.

   To enable SSL connections to Enterprise Console, generate a PEM-encoded SSL certificate or ensure that you have access to an existing PEM-encoded certificate. Save the `.crt` and `.key` files to `/etc/pki/raas/certs`.

3. Update RaaS Configuration

   Open `/etc/raas/raas` in a text editor and configure the following values (replace `<filename>` with your certificate filename).

```
tls_crt: /etc/pki/raas/certs/<filename>.crt
tls_key: /etc/pki/raas/certs/<filename>.key
port: 443

sql:
  ssl: True
```

4. Restart the Enterprise API service.

```
$ sudo systemctl restart raas
```

5. Verify the Enterprise API is running.

```
$ sudo systemctl status raas
```

6. Confirm that you can connect to the web console in a web browser.

   - url: `https://your_raas_server/`
   - Username: `root`
   - Password: `salt`

Install Salt Master plugin

1. Log in to your Salt Master.
2. Download one of the Salt Master plugin Egg files based on the version of Python installed on the Salt Master (`python --version` from the command line).
3. Install the plugin (requires Python setuptools).

```
$ sudo easy_install-2.7 SSEAPE-5.5.0-py2.7.egg
```

   or

```
$ sudo easy_install-2.6 SSEAPE-5.5.0-py2.6.egg
```

4. Verify the `/etc/salt/master.d` directory exists. If it doesn't, create it.
5. Generate the master configuration settings.

```
$ sudo sseapi-config --all > /etc/salt/master.d/raas.conf
```

6. Edit the generated `raas.conf` file to update the following values:

   - **sseapi_ssl_validate_cert** - Validates the certificate that Enterprise API uses. The default is `True`. If you are using your own CA-issued certificates, set this value to `True` and configure the `sseapi_ssl_ca`, `sseapi_ssl_cert`, and `sseapi_ssl_cert:` settings. Otherwise set this to `False` to not validate the certificate.

```
sseapi_ssl_validate_cert: False
```

- **sseapi_ssl_ca** - The path to a CA file.
- **sseapi_ssl_cert** - The path to the certificate. The default value is `/etc/pki/raas/certs/localhost.crt`.
- **sseapi_ssl_key** - The path to the certificate's private key. The default value is `/etc/pki/raas/certs/localhost.key`.
- **id** - Comment this line out by adding a `#` at the beginning. It is not required.
- **sseapi_server** - HTTP IP address of of your SaltStack Enterprise server, for example, `http://192.168.57.24`, or `https://192.168.57.24` if SSL is enabled.

7. Restart the Salt Master.

```
$ sudo systemctl restart salt-master
```

After a minute or two the Salt Master and its Minions appear in Enterprise Console.

Manually adding preloaded formulas (optional)

SaltStack Enterprise is distributed with several sample jobs and formulas. See Manually adding preloaded formulas (Jobs) when doing a manual installation of SaltStack Enterprise for instructions.

Upgrading SaltStack Enterprise

The following information will help you prepare for the SaltStack Enterprise upgrade.

Redis added to SaltStack Enterprise

Redis has been added to SaltStack Enterprise. You will need to install Redis 4.x into your environment. If your SaltStack Enterprise deployment is on a single host, install Redis. If your deployment is on multiple hosts, create a host for Redis.

1. Download the `Redis` and `jemalloc` installation packages for RH/CentOS that are provided under the *Manual installation* section.

   ```
   $ sudo yum install redis40u-4.0.11-1.ius.el7.x86_64.rpm jemalloc-3.6.0-1.el7.x86_64.rpm
   ```

2. Update configuration (if Redis is on a separate host)

   If you are setting up Redis on a host that is separate from SaltStack Enterprise, you will need to configure Redis to accept remote connections and to limit access using a password. To do this, update the `/etc/redis.conf` file, specifying the bind parameter and setting the password that your SaltStack Enterprise servers should use to authenticate.

   ```
   bind 0.0.0.0
   requirepass
   ```

3. Start the redis service

   ```
   systemctl enable redis
   systemctl start redis
   ```

PyCrypto extension required for raas Database

The `pycrypto` extension is now required. To make this available to your PostgreSQL database, complete the following steps.

1. Install the `postgresql-contrib` package on the PostgreSQL server.

   This extension requires PostgreSQL superuser privileges to install into the raas database.

   ```
   $ sudo yum install postgresql-contrib
   ```

2. Grant the raas user superuser privileges:

   The `pycrypto` extension requires Postgres superuser privilege to install into the raas database.

   ```
   $ su - postgres
   $ psql -d raas_<your_customer_id> -c 'ALTER USER <your_user> WITH SUPERUSER'
   ```

3. Perform the raas upgrade. See Upgrading.

4. Revoke the superuser privilege from the raas user.

   ```
   $ psql -d raas_<your_customer_id> -c 'ALTER USER <your_user> WITH NOSUPERUSER'
   ```

What to back up prior to the upgrade

- **Files / Directories**
  - `/etc/raas/raas`
  - `/etc/raas/pki/` - this contains hidden files, so back up the entire directory
  - `/etc/salt/master.d/raas.conf` - located on each Salt Master
  - `postgres.conf` - if local PostgreSQL
  - `pg_hba.conf` - if local PostgreSQL
- **Database**

  When upgrading your raas server, the database schema will need to be updated. Make sure to create a backup of your database before the upgrade. An example of this command is:

```
pg_dump -U salt_eapi raas_abc123... > postgres_raas_backup_$(date +%Y-%m-%d).sql
```

- **Targets, jobs, and the default SaltStack Enterprise Filesystem**

  During the upgrade process, the default targets, jobs, and files in the Filesystem are updated to the latest version. Modifications you've made to these files are not preserved.

  Note the following:

  - Pillar assignments to *default* targets are not preserved. These need to be re-assigned after upgrade.
  - Pillar assignments to *custom* targets are preserved.
  - Custom files, pillar data, and jobs are preserved.
  - Role customizations are preserved.
  - User accounts and passwords are preserved.

Upgrading

1. **Back up your database**. See PostgreSQL database backups for more information.
2. Save any changes you made to the default file system, pillar data, and jobs as new files or jobs.
3. Note any pillar assignments that are made to the default targets. These need to be re-assigned after upgrade.

On the SaltStack Enterprise server

1. Stop the `raas` service.

   ```
   $ sudo systemctl stop raas
   ```

2. Back up or remove the log file(s) at `/var/log/raas/raas`. This provides a clean log file if troubleshooting is required.
3. Back up or rename the `/etc/raas/raas` config file. You'll need to restore this file after upgrading.
4. Remove the currently installed version of Enterprise API.

   ```
   $ sudo yum remove raas
   ```

5. Upgrade SaltStack Enterprise server by installing the latest RPM. For example:

   ```
   $ sudo yum install raas-5.5-0.el7.x86_64.rpm
   ```

6. Restore the backup of the `/etc/raas/raas` config file.
7. Add the following new configuration options to `/etc/raas/raas`:

   - Change `driver: pg8000` to `driver: psycopg2`.

   ```
   background_workers:
     combined_process: True
     max_tasks: 100000
     max_memory: 1048576

   redis:
     url: redis://<Redis_IP>:6379
   ```

```
# AD/LDAP driver configuration
#authers:
#  ldap:
#    # Allowed log levels
#    # - OFF       # nothing is logged
#    # - ERROR     # only exceptions are logged
#    # - BASIC     # library activity is logged, only operation result is shown
#    # - PROTOCOL  # LDAPv3 operations are logged, sent requests and received responses are shown
#    # - NETWORK   # socket activity is logged
#    # - EXTENDED  # LDAP messages are decoded and properly printed
#    log_level: ERROR
#    ssl:
#      # Allowed ciphers values
#      # - TLSv1.2
#      # - SSLv23  # Default value selected if not defined when the
#      #           # AD auth config has use_ssl set to True
#      ciphers: SSLv23
#      ca_cert: /path/to/ca_cert.pem
#      client_key = /path/to/client.key
#      client_cert = /path/to/client.pem
```

8. Start the raas service in debug mode in the foreground with:

```
$ sudo raas -l debug
```

> This allows you to confirm that the `/etc/raas/raas` file was updated correctly. The process exits with a note that an update to the raas database is required. If you see this message, proceed with the upgrade. Otherwise, review your `raas.conf` file.

9. Upgrade the raas database with:

```
$ sudo raas upgrade
```

> Depending on the size of your database, the upgrade can take anywhere from several minutes to over an hour.

If you encounter errors, check the `/var/log/raas/raas` logfile for more information.

10. Start the Enterprise API service.

```
$ sudo systemctl enable raas
$ sudo systemctl start raas
```

11. Upgrade the “Preloaded Formulas”.

To upgrade the “Preloaded Formulas” (Jobs), see KB Manually Adding “Preloaded Formulas” (Jobs) when doing a Manual Installation of SaltStack Enterprise .

On each connected Salt Master

1. Stop the `salt-master` service.

```
$ sudo systemctl stop salt-master
```

2. Delete the SSEAPI Python module:

Delete the prior version of the SSEAPE module (this is the SaltStack Enterprise plugin for the Salt Master). For example:

```
RHEL:
$ sudo rm -rf /usr/lib/python2.7/site-packages/SSEAPE*

Ubuntu:
$ sudo rm /usr/lib/python2.7/dist-packages/SSEAPE*
```

3. Manually upgrade the Salt Master plugin by installing the updated Python egg.

```
$ sudo easy_install-2.7 SSEAPE-5.5.0-py2.7.egg
```

4. Update the eAPI Master paths.
    - Edit `/etc/salt/master.d/eAPIMasterPaths.conf` to reference the path to the new egg version.
    - If you are upgrading from a version of SaltStack Enterprise prior to 5.4, use the following command to generate the the paths:

      ```
      $ sudo sseapi-config --ext-modules > /etc/salt/master.d/eAPIMasterPaths.conf
      ```

    - Be sure to remove any of the path references from `/etc/salt/master.d/raas.conf`, as these were relocated in a previous release of SaltStack Enterprise.
5. Add `- job_completion: {}` to the engines section of `/etc/salt/master.d/raas.conf`.
6. Start the `salt-master` service.

    ```
    $ sudo systemctl start salt-master
    ```

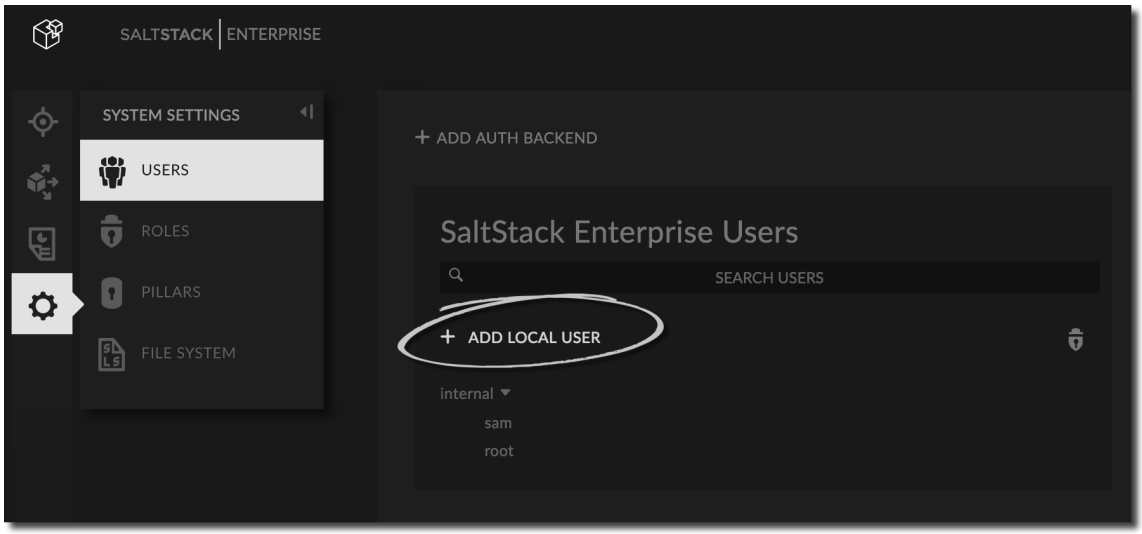
Initial configuration

- Create credentials for Salt Masters
- Change the root password
- Enable more accurate presence detection
- Back up critical data

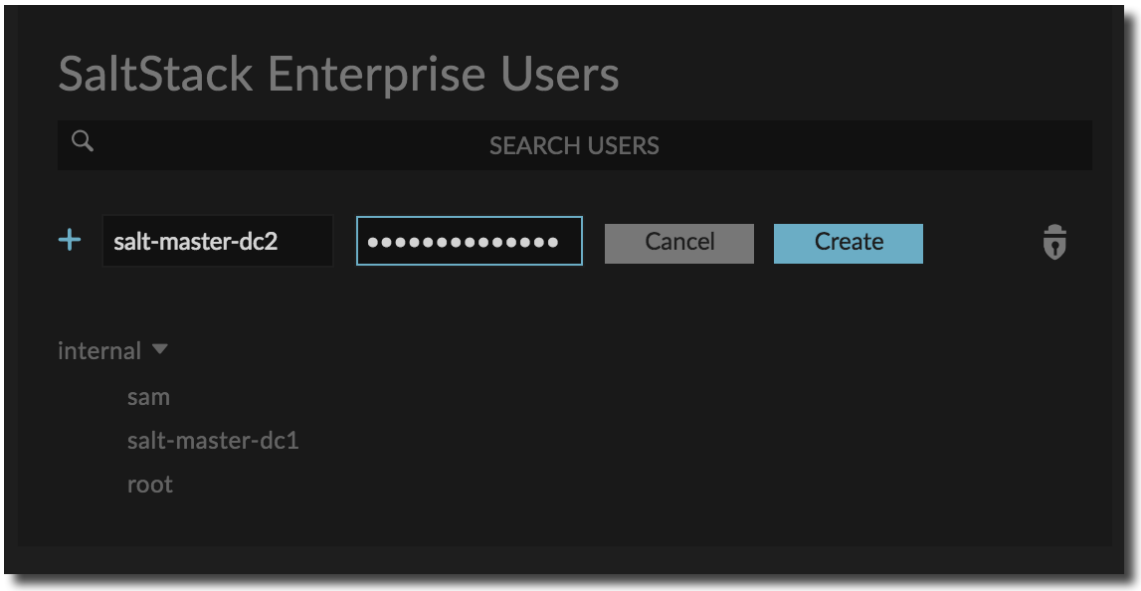
Create credentials for Salt Masters

Each Salt Master is configured to use the superuser account to connect to Enterprise API to simplify initial installation. For increased security, you should generate an Enterprise API account for each Salt Master.

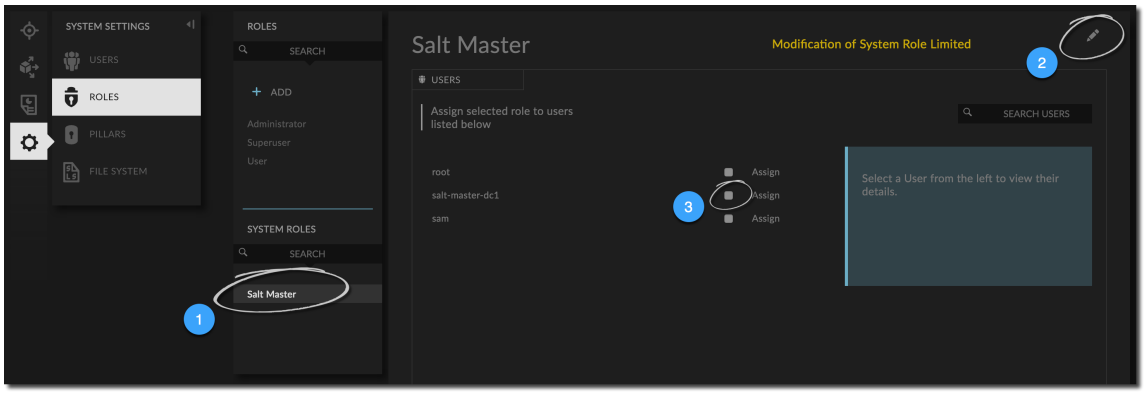
1. Open Enterprise Console and log in using the superuser account.
2. Click **System Settings** > **Users**.



3. Create a user account for each Salt Master.



4. Click **System Settings** > **Roles**, and select **Salt Master**. Click **Edit** and add each Salt Master account to the Salt Master system role.



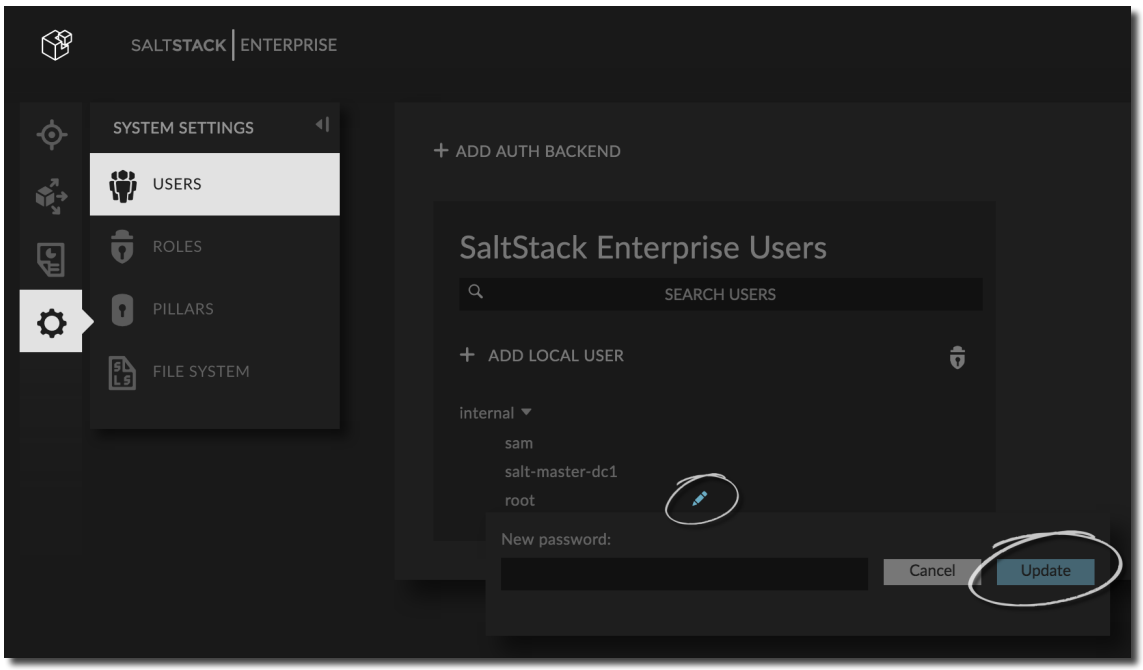
5. On each Salt Master, edit the `/etc/salt/master.d/raas.conf` file and update the `sseapi_username` and `sseapi_password` with the account credentials you created.
6. Restart the Salt Master service.

```
$ sudo systemctl restart salt-master
```

Change the root password

You can change the default password for the root user.

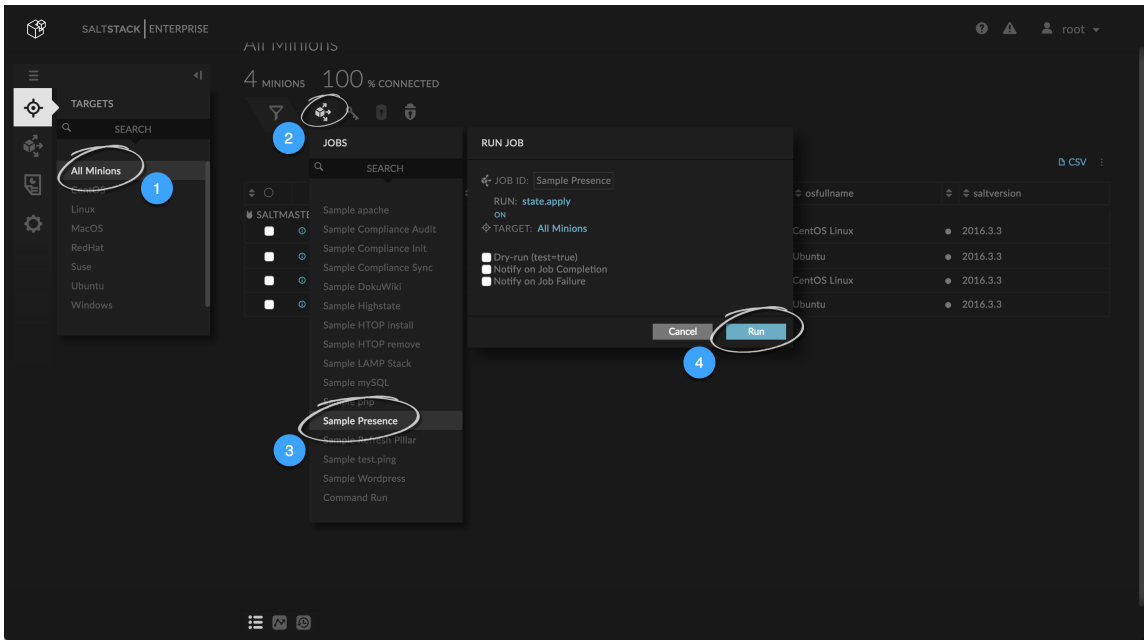
1. Open Enterprise Console and log in using the superuser account.
2. Click **System Settings** > **Users**.
3. Click the Edit icon next to the root account and provide a new password.



Enable more accurate presence detection

SaltStack Enterprise provides a job to install a Salt Beacon that sends periodic heartbeats from each Salt Minion. We recommend running this job on all minions to enable more accurate presence.

1. Open Enterprise Console and log in using the superuser account.
2. Click **Targets** > **All Minions** > **Jobs** > **Enable Presence**.
3. Click **Run**.



Back up critical data

If you are not using a complete system backup solution that can restore your entire SaltStack Enterprise server, at a minimum you should back up the following files:

- **/etc/raas/pki** - This directory contains a hidden file named `raas.key` that is used to encrypt the pillar data while at rest in the database. If you need to restore your SaltStack Enterprise server by re-installing, it is critical that you restore the original `raas.key` used when the database was created. If this file is lost, you lose all pillar data values in the Enterprise API file system.
- **/etc/raas/raas** - This file contains SaltStack Enterprise configuration data.
- **Enterprise API Database** - Configure regular PostgreSQL database backups for the Enterprise API database.

Congratulations!

You are now ready to manage your infrastructure using SaltStack Enterprise. Click the help icon in Enterprise Console for additional guidance.

Agentless Windows module

Download

Download the agentless Windows module files on the SaltStack Enterprise website.

Requirements

- English version of Windows
- Windows versions:
  - Windows 7
  - Windows 8.1
  - Windows 10
  - Windows Server 2008 R2
  - Windows Server 2012 R2
  - Windows Server 2016
- Powershell 3.0 or later
- WinRM must be configured and running
- The `/etc/salt/roster` file must have a configuration section for every Windows machine you want to connect to. The configuration must have a local admin user and password for each machine, as in the following example.

```
win2012dev:   # Minion ID
  host: <IP address>
  user: <local Windows admin username>
  passwd: <password for the admin user>
  winrm: True
```

> Domain credentials are not supported.

- Python 2 must be installed on the Salt Master. The `salt-ssh` module for Windows is supported only on Python 2, not later versions.
- pip 2 must be installed.
  - CentOS 7

```
$ yum install epel-release -y
$ yum install python-pip
$ pip install -U setuptools
```

  - Ubuntu 18.04

```
$ apt-get install python-pip
```

Installing the agentless Windows module

1. Use pip to install the `whl` file.

```
$ pip install -U ./saltwinshell-2017.7-cp27-cp27mu-linux_x86_64.whl
```

2. Edit `/etc/salt/roster` with your minion information.

```
testwin:   # Minion ID
  host: <IP address>
  user: <local Windows admin username>
  passwd: <password for the admin user>
  winrm: True
```

You can now run any Salt SSH command on the Windows server, such as the following:

```
$ salt-ssh testwin disk.usage
```