Date: 23.07.2025



Copyright @ 2024 PibyThree.com All Rights Reserved

SaltStack Master-Minion Architecture & Service Management

Contents

[1. Setting up Salt Master & Minions 2](#_Toc204941342)

[2. Accepting minion keys 2](#_Toc204941343)

[3. Running Salt Commands 3](#_Toc204941344)

[Test the connection 3](#_Toc204941345)

[Working with Services 3](#_Toc204941346)

[Working with Grains 3](#_Toc204941347)

# 1. Setting up Salt Master & Minions

First, we require 2 instances of Amazon Linux 2023 AMI. One for salt-master and another for salt-minion. We also require an instance of Microsoft Windows Server 2025 Base AMI for salt-minion.

The task of EC2 provisioning can be automated with `ansible-playbook`. Ensure you have `amazon.aws` collection installed with `ansible-galaxy collection list`. If not, run `ansible-galaxy collection install amazon.aws`.

Also ensure you have `aws cli` preconfigured before running the playbooks. If not, create access key from AWS console and run `aws configure` on the terminal.

Git Repository for the playbook: [aryan-madhavi/ansible-saltstack](https://github.com/aryan-madhavi/ansible-saltstack.git)

Run the playbook with: `ansible-playbook -i hosts launch-salt-cluster.yml`.  
Use the `-v` flag if required.

Note:

* Config file for master and minion services are stored in `/etc/salt/master` and `/etc/salt/minion` respectively in Linux.
* By default salt-master searches for state files in `/srv/salt/` in Linux.
* Config file for minion service is stored in `C://Program Data/Salt Project/Salt/conf/minion` for default installation in Windows.

# 2. Accepting minion keys

SSH into the salt-master instance and run the following commands:  
 $ sudo su -

$ salt-key -L

Now, you will see a list of unaccepted keys. One from the linux minion and another from windows minion. Accept them by running the following command for each key:

$ salt-key -a ‘key-id’

You may also accept all unaccepted keys with a single command:

$ salt-key -A

The salt master-minion architecture has been successfully setup.

# 3. Running Salt Commands

Test the connection between the master and the minions.

$ salt ‘\*’ test.ping

## Working with Services

|  |  |
| --- | --- |
| **Description** | **Command** |
| Start | salt ‘<minion-id>’ service.start <srvc\_name> |
| Stop | salt ‘<minion-id>’ service.stop <srvc\_name> |
| Restart | salt ‘<minion-id>’ service.restart <srvc\_name> |
| Status (Running?) | salt ‘<minion-id>’ service.status <srvc\_name> |
| List | salt ‘<minion-id>’ service.get\_all <srvc\_name> |
| Detailed Info | salt ‘<minion-id>’ service.info <srvc\_name> |

## Working with Grains

|  |  |
| --- | --- |
| **Description** | **Command** |
| List minion ids | salt ‘<minion-id>’ grains.item |
| List all grains | salt ‘<minion-id>’ grains.items |
| Get specific grain | salt ‘<minion-id>’ grains.get <grain> |