

# Implementing SaltStack for Configuration Management

---



**Andrew Mallett**

LINUX AUTHOR AND TRAINER

@theurbanpenguin [www.theurbanpenguin.com](http://www.theurbanpenguin.com)



# Objectives



## Understanding SaltStack

- SaltStack Project
  - VMware vRealize SaltStack Config
  - Salt Open
- Server / Client
  - TCP port 4505/4506 on server
  - Very scalable
- Python Based
  - Using YAML state files
  - Uses the .sls suffix



# Salt Repos

**openSUSE does ship with Salt: so  
we will use openSUSE**



**Ubuntu 22.04 ships with the  
latest version of Salt but 20.04  
does not have Salt at all**



# Salt

Although Salt is client-server based, like Puppet it can be used in a stand-alone format. The command `salt` is used on the Salt-Master to publish jobs to Salt Minions; whereas the command `salt-call --local` is used in a Masterless environment on the Salt-Minion.



```
$ sudo zypper install salt-minion  
$ sudo salt-call --local --version  
$ sudo salt-call --local test.ping  
$ sudo salt-call --local test.versions
```

## Working on openSUSE 15.2

CompTIA recommend learning Ubuntu, SUSE and RedHat based distributions for the Linux+ exams. openSUSE has Salt in the repositories by default so it makes sense to use this. When SUSE was owned by Novell, they were based based in Utah. SUSE still uses Salt in its Service Manager product



# Demo



## Working on the openSUSE System:

- Install the salt-minion
- Check version
- Run simple remote execution commands



# Salt Modules - Remote Execution/State

## Remote Execution Modules

Like ad-hoc commands in  
Ansible and Puppet

`pkg.install`

## State Modules

Like Puppet manifests or  
Ansible Playbooks

`pkg.installed`



```
$ sudo salt-call --local sys.list_modules  
$ sudo salt-call --local sys.list_functions pkg  
$ sudo salt-call --local sys.doc pkg.install
```

## Documentation

As well as web documentation, Salt ships with extensive command line documentation for the modules and functions. Here, we list help for the Remote Execution modules.






# Demo



## Remote Execution Documentation

- List modules
- List functions
- Add host entries





```
elif _operation == "MIRROR_Y":
    mirror_mod.use_x = False
    mirror_mod.use_y = True
    mirror_mod.use_z = False
elif _operation == "MIRROR_Z":
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

#selection at the end -add back the deselected mirror modifier
mirror_ob.select= 1
modifier_ob.select=1
bpy.context.scene.objects.active = modifier_ob
print("Selected" + str(modifier_ob)) # modifier ob is the active ob
#mirror_ob.select = 0
#one = bpy.context.selected_objects[0]
#bpy.data.objects[one.name].select = 1
#bpy.data.objects[one.name].select = 1
```

## Modules and State Files

- A different set of modules are used with State files
- State files allow for the desired state of a system to be declared
- By default, these state files should be created in the `/srv/salt` directory and use `.sls` extension



```
$ sudo mkdir /srv/salt
$ echo "set bg=dark modeline" >> ~/.vimrc
$ echo "# vim: set ft=yaml ts=2 ai sw=2 et cul cuc :" > common.sls
$ vim common.sls
```

## Your First State File

A state file defines the desired state of the system. We can install multiple packages should be need. The vim-data package is required for syntax highlighting,



common.sls

```
# vim: set ft=yaml ts=2 ai sw=2 et cul cul :
```

```
install_common_packages:
```

```
  pkg.installed:
```

- pkgs:
  - rsync
  - vim-data
  - curl

Configure UK Time Zone on Salt Minions:

```
  timezone.system:
```

- utc: True
- name: Europe/London

```
$ sudo cp common.sls /srv/salt
$ sudo salt-call --local state.sls common test=True
$ sudo salt-call --local state.sls common
```

## Apply State File

The file needs to be copied to the `/srv/salt` directory and is executed using the `state.sls` Remote Execution module. If needed, we can test before the full operation.



# Demo



## Working with Salt States:

- Modify vim modeline
- Create state file
- Apply state file



# Summary



## Understanding SaltStack from VMware

- Client / Server model
- Inbound TCP 4505:4506 on server
- Local only with salt-call --local
- File root: /srv/salt
- Salt state files based on YAML



## Implementing Chef for Configuration Management

\* \* \* \* \*

A close-up photograph of a person's hands typing on a laptop keyboard. The keyboard has white keys on a dark background. A semi-transparent blue rectangular overlay is positioned over the hands and keyboard. Inside this overlay, there is a white rounded rectangle containing a row of ten white asterisks (\* \* \* \* \*). The background is dark and out of focus, showing the person's arms and the laptop screen.