# Network Orchestra

*Release 1.0*

Jan 02, 2020

# Contents:

You will see amazing things that can be done with Ansible and Salt in the OpenSource environment. The community out there is amazing... Enjoy!

Project code at https://github.com/aegiacometti/netor

Project documentation at https://netor.readthedocs.io/en/latest/index.html

CHAPTER 1

# Introduction

## 1.1 Purpose of the project

This is a very simple compilation of several OpenSource packages, which by using scripts custom scripts help to start the journey of network automation and orchestration, without having to learn from the very beginning how to configure all of them.

Why? Because I believe that the most important factor in the adoption of any new methodology is to make it easier to start using them.

So, the scripts won't be nice, they are very simple and they work fine, in fact, they are very easy to read. They catch some typing errors but pay attention when you write. Anyway, do not worry, you won't break anything at this point.

The tools that integrate at the moment are:

- Ansible
- Salt
- TinyDB
- Slack

As I move forward I will try to integrate other packages and functionalities.

This project is only to help start using Ansible and Salt, in order to see what you can get out of them, and after that, you should start learning about those two projects which are amazing.

## 1.2 Motivation

After trying several network tools that claim to be essentials to networking, as you already may know, there is no tool that will really work as you need, or even as they claim. They may be useful for some tasks, but at some point, I always ended quitting after hours of trying and talking with the official support. Every tool is beautiful in the PPTs and demos, but then when you deploy it is when the adventure starts.

Having real support from providers is so slow and even sometimes you don't have time to wait and you do it on your own. (THIS TOOL WILL NOT BE DIFFERENT, but at least is fun to learn and develop in an OpenSource environment) :)

A couple of years ago I started to learn Python and I love it, I used it to create a couple of scripts that helped me a lot to support several network-related projects deployments.

Later on, I learned about Ansible and it was "wow" I really love this!

In the path of learning network automation and orchestration with Ansible and Salt, I found my self having to configure different files in different locations and in a completely different manner. There are great tutorials on how to lear and use them, but nothing to help you to integrate them, in order to make it easier to start experimenting.

Because to be honest, both Ansible and Salt are great. Ansible is simple to start using it for simple things, but Salt from my point of view is incredible, has similar functions and a lot of very cool capabilities but at the beginning it is hard to start.

Last but not least, add an integration with Slack, now you can "chat" with your infrastructure!

So, as I love to learn and to build things, I decided to start this adventure of learning and develop a personal tool using Python, in an OpenSource manner.

CHAPTER 2

# How to install

## 2.1 Download the VM

The easiest way is to use the VM from this link:

https://drive.google.com/open?id=14p02l8FkTLCYX_fkISyDAcEt-v7igCjU

With 2Gb of RAM, and 2 processor it should be OK.

userID: netor password: password (change it! with `passwd`)

Ansible do not require much power, is supper light, Salt is more less the same, except if you use Proxy Minions

The Salt proxy minion processes uses a little bit of power/memory.

I recommend you to read a little bit about Salt proxy minion, what they are and why they exist in the networking environment. And later about **salt-sproxy**, a proxy-less approach. Follow Mircea Ulinic, this guy is a genius.

## 2.2 Linux bash installer

This is a very simple copy&paste of the commands below in the format of a bash script. Download it with in the directory you want to install Netor and it will create the home directory and install the packages:

```
wget https://raw.githubusercontent.com/aegiacometti/netor/master/bin/
netor-install.sh
```

```
bash netor-install.sh
```

Finally with root privileges add the following environment variable and restart the system.

```
`NETOR="/home/netor/netor/"`
```

Or follow the below instruction to go step by step with the commands.

## 2.3 Installation on Linux

**Requirements**

Read about these packages:

- setuptools
- python3
- tinydb
- ansible
- ansible network engine role
- salt (using the bootstrap installed, details below)
- salt-sproxy
- slack-client

**Installation**

1. Update apt package list:

   ```
   sudo apt-get update
   ```

2. Install pip3 (we will use Python3):

   ```
   sudo apt-get install python3-pip
   ```

3. Upgrade setuptools:

   ```
   sudo pip3 install setuptools --upgrade
   ```

4. Install TinyDB:

   ```
   sudo pip3 install tinydb
   ```

*For reference:* https://tinydb.readthedocs.io/en/latest/getting-started.html

5. Install git:

   ```
   sudo apt-get install git
   ```

6. Clone `netor` project repository:

Create a directory for the clone of the repository or do the clone directly at you home directory, this will be the project home:

```
git clone https://github.com/aegiacometti/netor.git
```

7. Install NAPALM:

   ```
   sudo pip3 install napalm
   ```

*For reference:* https://napalm.readthedocs.io/en/latest/

8. Install Ansible:

   ```
   sudo pip3 install ansible
   ```

   ```
   ansible-galaxy install ansible-network.network-engine
   ```

*For more detail refer to installation guides at:* https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html https://galaxy.ansible.com/ansible-network/network-engine

9. Install Python Windows driver:

```
`sudo pip3 install "pywinrm>=0.3.0"`
```

10. Install Python Slack module:

```
`sudo pip3 install slackclient==1.3.2`
```

11. Download a base and clean ansible.cfg and copy it into your $HOME directory from GitHub:

```
wget https://raw.githubusercontent.com/ansible/ansible/devel/examples/
ansible.cfg -O $HOME/.ansible.cfg
```

12. Now, for the first time, you have to configure netor by manually executing the python script:

```
python3 [netor_home_directory]/netor/tinydb/scripts/netorconf.py
```

In the future is you clone a new `netor` deployment for testing or to have 2 directories to work separately, you will have to do this procedure again.

13. Add to your PATH environment the `netor/bin` directory for easy execution of scripts:

```
vi $HOME/.profile
```

and add at the end as an example `PATH="$PATH:[netor_home_directory]/netor/bin/"`

14. Add the installation directory to the system environment:

```
`sudo vi /etc/environment`
```

Add the line at the end:

```
`NETOR="/home/adrian/netor-master/"`
```

15. Logoff session and login again:

If everything worked fine you can view the commands with tab autocomplete.

netor-db-list

netor-db-customer

etc

...

16. Install Salt:

The recommended way is to use the bootstrap:

```
wget -O bootstrap-salt.sh https://bootstrap.Salt.com
sudo sh bootstrap-salt.sh -x python3 -M
```

Now, Salt has a couple of bug which I corrected and ask to merge on the master repositories. Since this could take a while to refresh, download these 2 files to replace them in your PC:

https://raw.githubusercontent.com/aegiacometti/salt/master/salt/runners/bgp.py https://raw.githubusercontent.com/aegiacometti/salt/master/salt/runners/net.py

To find them on your PC use:

```
sudo find /usr -name net.py sudo find /usr -name bgp.py
```

The ones under a directory called `runners`

*For more information go to the project page, they have great documentation:* https://docs.saltstack.com/en/latest/topics/tutorials/walkthrough.html https://docs.saltstack.com/en/latest/topics/tutorials/walkthrough_macosx.html

Now, unlike Ansible, Salt uses daemons and the bootstrap add them to auto-start, and we don't want that, we want to start them manually, just in case to not have them running and searching for the devices when we don't want or when they are not even reachable, as an example, if we are at home, another customer, or in a meeting!

In order to stop them and then disable them from auto-start we need to execute this commands:

```
netor-salt-stop

sudo systemctl disable salt-master.service

sudo systemctl disable salt-minion.service

netor-salt-start
```

17. Copy Salt minion proxy to the systemd folder:

```
sudo cp [netor_home_dir]/netor/salt/config/services/salt-proxy@.service
/etc/systemd/system/
```

*(this path could vary depending on the system)*

18. Backup the original Salt master and minion configuration files (so you can have them as a reference), and create symbolic links to Salt new configuration files:

```
sudo mv /etc/salt/master /etc/salt/master.bkp

sudo mv /etc/salt/minion /etc/salt/minion.bkp

sudo ln -s [netor_home_dir]/netor/salt/config/master /etc/salt/master

sudo ln -s [netor_home_dir]/netor/salt/config/minion /etc/salt/minion

sudo ln -s [netor_home_dir]/netor/salt/config/proxy /etc/salt/proxy
```

19. Install salt-sproxy:

```
sudo pip3 install salt-sproxy
```

20. Run `netor-db-push` generate Ansible and Salt configuration files.

21. Restart Salt daemons:

```
netor-salt-restart
```

22. done!

## 2.4 Installation on MacOS

I have tested the software on Linux and Mac. I was able to install it on Mac.

Ansible works just fine, but Salt not so good, it is not officially supported, but since it is a kind of Unix, in this case FreeBSD, it work, but not quiet well. So bellow you have the install procedure, but if want to go for sure and you have a Mac, i would download the VM and after i know how Salt works, just then i would try Salt on Mac directly.

**Requirements**

Read about this packages:

- xcode-select developer
- homebrew
- python3
- ansible network engine role

- saltstack
- salt-sproxy
- slack-client

**Installation**

1. Install xcode-select for command line developer tools:

```
xcode-select --install
```

2. Install Homebrew package manager:

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/
Homebrew/install/master/install)"
```

3.- Install Python 3:

```
brew install python3
```

4. Install TinyDB:

```
sudo pip3 install tinydb
```

*For reference:* https://tinydb.readthedocs.io/en/latest/getting-started.html

6. Clone `netor` project repository:

Create a directory for the clone of the repository or do the clone directly at you home directory, this will be the project home:

```
git clone https://github.com/aegiacometti/netor.git
```

7. Install NAPALM:

```
sudo pip3 install napalm
```

*For reference:* https://napalm.readthedocs.io/en/latest/

8. Install Ansible:

```
sudo pip3 install ansible
```

```
ansible-galaxy install ansible-network.network-engine
```

*For more detail refer to installation guides at:* https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html https://galaxy.ansible.com/ansible-network/network-engine

9. Install Python Windows driver:

```
`sudo pip3 install "pywinrm>=0.3.0"`
```

11. Install Python Slack module:

```
`sudo pip3 install slackclient==1.3.2`
```

12. Download a base and clean ansible.cfg and copy it into your $HOME directory from GitHub:

```
curl https://raw.githubusercontent.com/ansible/ansible/devel/examples/
ansible.cfg -o $HOME/.ansible.cfg
```

13. Now, for the first time, you have to configure netor by manually executing the python script:

```
python3 [netor_home_directory]/netor/tinydb/scripts/netorconf.py
```

In the future is you clone a new `netor` deployment for testing or to have 2 directories to work separately, you will have to do this procedure again.

14. Add to your PATH environment the `netor/bin` directory for easy execution of scripts:

```
sudo nano /etc/paths
```

and add at the end `[netor_home_directory]/bin/`

If everything worked fine you can view the commands with tab autocomplete.

netor-db-list

netor-db-customer

etc

...

15. Install Salt:

```
brew install saltstack
```

Now, Salt has a couple of bug which I corrected and ask to merge on the master repositories. Since this could take a while to refresh, download these 2 files to replace them in your PC:

https://raw.githubusercontent.com/aegiacometti/salt/master/salt/runners/bgp.py https://raw.githubusercontent.com/aegiacometti/salt/master/salt/runners/net.py

To find them on your PC use:

```
sudo find /usr -name net.py sudo find /usr -name bgp.py
```

The ones under a directory called `runners`

*For more information go to the project page, they have great documentation:* https://docs.saltstack.com/en/latest/topics/tutorials/walkthrough.html https://docs.saltstack.com/en/latest/topics/tutorials/walkthrough_macosx.html

Now, we need to add the service files to launchd to be able to start the daemons:

```
sudo cp [full_netor_home_dir]/netor/salt/config/services/
com.saltstack.master.plist /Library/LaunchDaemons/             sudo cp
[full_netor_home_dir]/netor/salt/config/services/com.saltstack.
minion.plist /Library/LaunchDaemons/
```

And we will start or stop or restart them with:

```
netor-salt-start netor-salt-stop netor-salt-restart
```

16. Verify `maxfiles` parameters at OS level:

```
sudo launchctl limit
```

If they are lower than 100000, you will need to change this. Usually happens on old MacOS versions.

```
sudo cp [full_netor_home_dir]/netor/salt/config/services/limit.
maxfiles.plist /Library/LaunchDaemons
```

Adjust the values after the line `maxfiles`, add it to the launchd.

```
sudo launchctl load -w /Library/LaunchDaemons/limit.maxfiles.plist
```

Restart the computer for this change to take effect.

17. Salt master and minion configuration files:

For your reference you can find clean samples at `/user/local/etc/saltstack`

Create these links to use as defaults, these files will by the updated ones from Netor:

```
sudo ln -s [full_netor_home_dir]/netor/salt/config/master /etc/salt/
master
```

```
sudo ln -s [full_netor_home_dir]/netor/salt/config/minion /etc/salt/
minion
```

```
sudo ln -s [full_netor_home_dir]/netor/salt/config/proxy /etc/salt/
proxy
```

18. Install salt-sproxy:

    ```
    sudo pip3 install salt-sproxy
    ```

19. Run `netor-db-push` generate Ansible and Salt configuration files.

20. Restart Salt daemons:

    ```
    netor-salt-restart
    ```

21. done!

## 2.5 Updates

In order to update with the latest changes, just CD into your netor directory and pull the changes with:

```
git pull origin master
```

CHAPTER 3

# Important comments

Remember to create your own database and push it to Ansible and Salt, is super easy, follow the guide at **How to use** section.

All of this software is Open Source, which means that is free and community maintained.

There are a couple of security configuration settings from Ansible and Salt that are not recommended to use, but for learning purposes I let them open.

**For Ansible:**

File $HOME/.ansible.cfg

```
host_key_checking = False
```

File $HOME/.ansible.cfg

```
host_key_auto_add = True
```

For Salt:

File /etc/salt/master and /etc/salt/minion

```
open_mode:  True
```

```
auto_accept:  True
```

When you restart Salt, give a couple of minutes to synchronise, it will look like not working… wait… and if you want you can check the logs of the daemons with:

```
sudo tail -f /var/log/salt/*
```

CHAPTER 4

# How to use

## 4.1 Functionalities

The following sections will contain a review of how to use the scripts.

All the scripts are located at the /bin directory of your netor install directory or home directory.

They are written in BASH (i just wanted to try BASH :) ), but in the future and in order to make it easier to start using Ansible and Salt features, I will add some other scripts in python (so much easier :) ). And why adding those script to use Ansible and Salt? Because as you will see, lear the syntax of both is very very tedious when you are just starting to learn about them.

The main idea is to have an inventory DB, and then use that inventory to configure the inventories of Ansible and Salt. This would be useful because both configure their inventories in a completely different manner and using several different files.

As a quick example, Ansible in its basic form uses a "hosts" file and that it (of course there are more advanced ways for configuring its inventory), and Salt uses in its basic form uses several files, one top.sls and one .sls file per each device. So, as you can imagine this setup will be tedious to start using both tools at the same time to see what you can do in your learning journey.

## 4.2 First Steps

1. If you haven't done yet, run the script `python3 [netor_home_directory]/netor/tinydb/scripts/netorconf.py`. And verify that you have the $PATH environment variable set `echo $PATH` should show you your PATH to the Neto home directory.

You should be able to use the tab autocomplete. Try writing netor- and then press tab.

*(From now on you can use ''netor-config''for setting a new Netor home directory, since you should already have it in your $PATH)*

2. Now try to look at what do you have in the DB with `netor-db-list`. And where is located the filename of the DB.

3. You can try to list, add, edit and modify each of the DB tables (customers, sites and devices), by using the scripts `netor-db-customers`, `netor-db-sites` and `netor-db-devices`.

4. Once you have seen how it works, you can try to set up your own DB with `netor-db-switch`, and start adding data to it, always starting with customer -> site -> devices.

This script could be useful if you want to have different DBs for different purposes. Let's say, production and development, site 1 and site 2, core switches and access switches, routers, firewalls, project 1 and project 2, etc, etc, etc....

Use any combination that is useful to what you have to do.

5. Now push your new DB to Ansible and Salt inventories with `netor-db-push`.

When you do this the configuration files of Ansible and Salt will be completely replaced.

In this process, you will be able to choose if you want to push the whole DB, or use a filter to chose with devices you want to push, based on a RedEx expression. You will have to confirm so don't hesitate to play with the filer.

The push to Ansible will work right away, but for the push to Salt to take effect you will need to restart its daemons with `netor-salt-restart`

You also have available commands to stop and start Salt, with `netor-salt-stop` and `netor-salt-start`.

6. Since you could have devices with the same name at different sites, or at a different customer, the `netor-db-push` will generate hostnames bases on the join of the customer name, site name, and device name. This will assure the uniqueness of the hostname in Ansible and Salt.

And a very important feature is that you don't need to be able to resolve their FQDN names because it is very probable that you or the customer don't have them on the DNS, and if they do, you could end up with conflicting names between customers or sites.

So then, the next interesting step is using the commands `netor-ping` and `netor-traceroute`. This two commands will run using this new naming convention and without the need of having their names in your local machine hosts file or in a external DNS.

Just remember, the netor hostname to use with these two commands, will be the conjunction of customer, site, and device using an underscore "_".

Let's make it easy with an example:

```
nadrian@adrian-VirtualBox:~$ netor-db-list

Using default DB File: /home/adrian/netor-master/netor/tinydb/data/db.json

LIST DATABASE

...

List Devices

Customer Name          Site Name              Device Name          Device IP
↪Device OS   User Name            Password             Salt Proxy Req
c1                     s1                     co-1                 10.100.12.2
↪ios         cisco                cisco                y
c1                     s1                     cpe                  10.0.12.2
↪ios         cisco                cisco                y
c1                     s1                     ua-1                 10.100.200.2
↪ios         cisco                cisco                y

Full DB listed: /home/adrian/netor/netor/tinydb/data/db.json
```

(continues on next page)

(continued from previous page)

```
adrian@adrian-VirtualBox:~$ netor-ping c1_s1_cpe
/home/adrian/netor-master/bin/netor-ping: line 8: 10.0.12.2: command not found
PING 10.0.12.2 (10.0.12.2) 56(84) bytes of data.
64 bytes from 10.0.12.2: icmp_seq=1 ttl=253 time=28.9 ms
64 bytes from 10.0.12.2: icmp_seq=2 ttl=253 time=30.1 ms
```

7. At some point, you could want to import or export the DB to/from CSV format. The export could be useful to work with the DB in another program, and then import it again, or if you go to a customer, just ask for the inventory to work with in CSV format, and then import it to Netor.

The respectively commands are: `netor-db-export` and `netor-db-import`.

Only take into consideration this format that you need to respect:

```
customer,site,dev_name,dev_ip,os,userid,passwd,salt_proxy_required
```

8. All the netor scripts log on screen and also on a logging file located at `./netor/log`
9. Ansible and Salt make the device backups to their directories:

    `./netor/ansible/backup` `./netor/salt/backup`

I left them in different locations just to be able to see the differences in action.

10. You will see at the DB that each device has a last setting named "Salt Proxy Required".

This is a core feature of salt, it means that it will have a process in constant connection to the remote device. This will allow you to execute commands on it super fast since it doesn't require to go through the login process, since it is already connected, and on the other hand it keep in an internal DB/like cache all the facts, arps, IPs, and some others things about the device, and you don't even have to worry about installing and managing a DB software like any other tool in the market require. You will learn how cool it that about Salt and is a big difference with Ansible. In fact, Ansible is beautiful, but Salt take thing to another level.

Later on i will talk about the Salt event-bus, wow that is sooo cool too.

11. That is it. Now start "playing" with Ansible and Salt.
12. If you are going to use Slack, you will need to crate a bot and add use the webhook token in the Salt master configuration file and Ansible Playbooks.

# CHAPTER 5

# Use cases

Now, imagine if from your PC at your work, or when you go to support one of your customers, in a couple of minutes you can perform all of the following basics out of the box procedures, they will see you as a hero!

You could even redirect the logging of several network devices to your PC while you are working in order to see them with a syslog server on your PC, or only with the Salt event-bus, and after the jobs is done, revert the syslog change in seconds.

All this examples are running against a GNS3 virtual lab in my PC, you can do the same with the VM or installing it following the install guides. I am only running legacy IOS but they support others OS via NAPALM, and of course, regular server operating system.

Some tips before we start:

**netor scripts**

In time i will create *netor*-x style script just as a mask of the following commands, and again, ir order to try to make it easier to start using Ansible ans Salt. And i will had a simple command to update the /bin folder.

**Ansible:**

- I am providing a couple of playbook, some parses, and scripts for you to experiment.
- You might see some warning related to Python2 getting to end of support, and some other pieces of code about to get deprecated. Typical linux style, letting you know about thing that are about to change.
- you con limit the devices to execute a playbook by adding `-l xxx`. Where xxx is regEx filter.
- you can add `--check --diff` to check commands before applying and to show the differences to apply.
- you can send information from the playbook to a parser to crop information and/or to script to do something else, this means that there is communication between to move information and act accordingly.
- `ansible-playbook` is the command to execute playbooks, you have to cd to the playbooks directory
- Ansible is kind of static, because it only do something when you enter a command. If you want to trigger actions you have to use an external tool, like Nagios. If Nagios detects something execute this Ansible playbook.

**Salt:**

- what it super cool about Salt is the even if the commands are weird at the beginning, all of the modules/functions have a help right at the command line, i will show you how.
- `salt` is for execution online commands against the devices
- `salt-run` is for executing commands with information that Salt already
- to every command you can add at the end `-l debug` to check what is going on.
- it has incredible `net.bgp` module to check for information, to configure and potentially react.
- Salt has a cache database with information gathered by runners, the cool thing about this is that you don't need to install and maintain a separate DB to store information like regular network management software requires.
- Salt has an event bus, which you can see with `salt-run state.event pretty=True`, the amazing thing about this is that you can start thinking in **Orchestration**, or in other words, define a reactor to an event when some message gets to the event bus. You can even attach a chat bot.

## 5.1 Ansible examples

From easy to hard

**Make a backup**

```
*adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook backup.
↪yml -l c1*

PLAY [Backup devices configs]␣
↪*********************************************************************************************

TASK [Read IOS configs]␣
↪*********************************************************************************************

ok: [c1_s1_cpe]

ok: [c1_s1_co-1]

ok: [c1_s1_ua-1]

TASK [Save IOS config]␣
↪*********************************************************************************************
changed: [c1_s1_cpe]
changed: [c1_s1_ua-1]
changed: [c1_s1_co-1]

PLAY RECAP␣
↪*********************************************************************************************
c1_s1_co-1                 : ok=2    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=2    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_ua-1                 : ok=2    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0

adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ls -la ../backup/
total 160
drwxr-xr-x 2 adrian adrian 4096 Nov 15 22:49 .
drwxr-xr-x 6 adrian adrian 4096 Nov 15 22:47 ..
```

(continues on next page)

(continued from previous page)

```
-rw-rw-r-- 1 adrian adrian 3428 Nov 15 22:49 show_run_c1_s1_co-1.txt
-rw-rw-r-- 1 adrian adrian 2262 Nov 15 22:49 show_run_c1_s1_cpe.txt
-rw-rw-r-- 1 adrian adrian 3082 Nov 15 22:49 show_run_c1_s1_ua-1.txt
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$
```

**show arp tables**

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook show-arp.
↪yml -l c1_s1_cpe

PLAY [Show IP ARP]␣
↪*********************************************************************************************

TASK [Show IP ARP]␣
↪*********************************************************************************************

ok: [c1_s1_cpe]

TASK [debug]␣
↪*********************************************************************************************
ok: [c1_s1_cpe] => {
    "list_of_ip_arp.stdout_lines": [
        [
            "Protocol  Address          Age (min)  Hardware Addr   Type   Interface",
            "Internet  10.0.12.1              68   c201.375c.0001  ARPA   ␣
↪FastEthernet0/0",
            "Internet  10.0.12.2               -   c202.5d80.0000  ARPA   ␣
↪FastEthernet0/0",
            "Internet  10.100.12.1             -   c202.5d80.0001  ARPA   ␣
↪FastEthernet0/1",
            "Internet  10.100.12.2            68   c204.5f8c.0000  ARPA   ␣
↪FastEthernet0/1"
        ]
    ]
}

PLAY RECAP␣
↪*********************************************************************************************
c1_s1_cpe                  : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

**gather-facts, which is the device basic information**

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook gather-
↪facts.yml -l c1_s1_cpe

PLAY [Gather IOS facts]␣
↪*********************************************************************************************

TASK [gather all facts]␣
↪*********************************************************************************************

ok: [c1_s1_cpe]

TASK [Display the OS version]␣
↪*********************************************************************************************
ok: [c1_s1_cpe] => {
```

(continues on next page)

(continued from previous page)

```
    "msg": "The hostname is r2 and the OS is 12.4(15)T13"
}

TASK [Display config]␣
↪*********************************************************************************************
ok: [c1_s1_cpe] => {
    "msg": {
        "ansible_facts": {
            "ansible_net_api": "cliconf",
            "ansible_net_config": "!\nversion 12.4\nno service pad\nservice tcp-
↪keepalives-in\nservice tcp-keepalives-out\nservice timestamps debug datetime msec␣
↪localtime show-timezone\nservice timestamps log datetime msec localtime show-
↪timezone\nservice password-encryption\n!\nhostname r2\n!\nboot-start-marker\nboot-
↪end-marker\n!\nlogging buffered 32000\nno logging console\nenable secret 5 $1$QAh2
↪$FiUShFDsaikloAgWmKsW1.\n!\naaa new-model\n!\n!\naaa authentication login default␣
↪local-case\naaa authorization exec default local \n!\n!\naaa session-id␣
↪common\nmemory-size iomem 5\nno ip source-route\nip options drop\nip cef\n!\n!\nip␣
↪dhcp bootp ignore\n!\n!\nno ip domain lookup\nip domain name quadrant.edu\n!
↪\nmultilink bundle-name authenticated\n!\n!\n!\n!\n!\n!\n!\n!\n!\n!\n!\n!\n!\n!\n!
↪\n!\n!\n!\n!\n!\n!\nfile prompt quiet\nusername cisco privilege 15 secret 5 $1$OKM5
↪$WoIzwQQ6Xrlt3ymrIH8VE/\narchive\n log config\n  hidekeys\n! \n!\n!\n!\nip ssh␣
↪version 2\nip scp server enable\n!\n!\n!\n!\ninterface FastEthernet0/0\n␣
↪description to_r1\n ip address 10.0.12.2 255.255.255.0\n no ip redirects\n no ip␣
↪proxy-arp\n duplex auto\n speed auto\n!\ninterface FastEthernet0/1\n description to_
↪inside\n ip address 10.100.12.1 255.255.255.0\n no ip redirects\n no ip proxy-arp\n␣
↪duplex auto\n speed auto\n!\ninterface FastEthernet1/0\n no ip address\n shutdown\n␣
↪duplex auto\n speed auto\n!\nrouter eigrp 1\n network 10.0.0.0\n no auto-summary\n!
↪\nip forward-protocol nd\nip route 0.0.0.0 0.0.0.0 10.0.12.1\n!\n!\nno ip http␣
↪server\nno ip http secure-server\n!\nip sla 1\n udp-echo 10.0.12.1 999\n timeout␣
↪4000\n tag probe1_test2\n frequency 5\n history lives-kept 1\n history buckets-kept␣
↪3\n history filter all\nip sla 2\n icmp-echo 10.0.12.1\n tag probe1_test1\n history␣
↪lives-kept 1\n history filter all\nsnmp-server community snmpCommunity RW\nsnmp-
↪server community read_only RO\nsnmp-server community read_write RW\n!\n!\n!\n!\n!\n!
↪\ncontrol-plane\n!\n!\n!\n!\n!\n!\n!\n!\nbanner login ^C\n\nUnauthorized access␣
↪is prohibited!\n\n^C\n!\nline con 0\n exec-timeout 20 0\n logging synchronous\nline␣
↪aux 0\n exec-timeout 0 1\n no exec\n transport output none\nline vty 0 4\n exec-
↪timeout 20 0\n logging synchronous\n transport input ssh\n transport output␣
↪ssh\nline vty 5 15\n exec-timeout 20 0\n logging synchronous\n transport input␣
↪ssh\n transport output ssh\n!\nntp server 10.0.0.2\n!\nend",
            "ansible_net_gather_network_resources": [],
            "ansible_net_gather_subset": [
                "default",
                "config"
            ],
            "ansible_net_hostname": "r2",
            "ansible_net_image": "tftp://255.255.255.255/unknown",
            "ansible_net_iostype": "IOS",
            "ansible_net_model": "3725",
            "ansible_net_python_version": "2.7.15+",
            "ansible_net_serialnum": "FTX0945W0MY",
            "ansible_net_system": "ios",
            "ansible_net_version": "12.4(15)T13",
            "ansible_network_resources": {},
            "discovered_interpreter_python": "/usr/bin/python"
        },
        "changed": false,
        "failed": false,
```

(continues on next page)

(continued from previous page)

```
        "warnings": [
            "default value for \`gather_subset` will be changed to \`min` from \`!
↪config` v2.11 onwards",
            "Platform linux on host c1_s1_cpe is using the discovered Python␣
↪interpreter at /usr/bin/python, but future installation of another Python␣
↪interpreter could change this. See https://docs.ansible.com/ansible/2.9/reference_
↪appendices/interpreter_discovery.html for more information."
        ]
    }
}

PLAY RECAP␣
↪*********************************************************************************************************
c1_s1_cpe                  : ok=3    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

**add a regular show command at 'cmd='**

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook ios-show-
↪cmd.yml -e cmd="'run | inc snmp'" -l c1_s1

PLAY [IOS show cmd]␣
↪*********************************************************************************************************

TASK [IOS show cmd]␣
↪*********************************************************************************************************

ok: [c1_s1_cpe]

ok: [c1_s1_co-1]

ok: [c1_s1_ua-1]

TASK [debug]␣
↪*********************************************************************************************************
ok: [c1_s1_co-1] => {
    "output.stdout_lines": [
        [
            "snmp-server community snmpCommunity RW"
        ]
    ]
}
ok: [c1_s1_ua-1] => {
    "output.stdout_lines": [
        [
            "snmp-server community snmpCommunity RW"
        ]
    ]
}
ok: [c1_s1_cpe] => {
    "output.stdout_lines": [
        [
            "snmp-server community snmpCommunity RW",
            "snmp-server community read_only RO",
            "snmp-server community read_write RW"
        ]
    ]
```

(continues on next page)

(continued from previous page)

```
}

PLAY RECAP␣
↪*********************************************************************************************************
c1_s1_co-1                 : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_ua-1                 : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook ios-show-
↪cmd.yml -e "cmd='ip int bri'" -l c1_s1

PLAY [IOS show cmd]␣
↪*********************************************************************************************************

TASK [IOS show cmd]␣
↪*********************************************************************************************************

ok: [c1_s1_cpe]

ok: [c1_s1_co-1]

ok: [c1_s1_ua-1]

TASK [debug]␣
↪*********************************************************************************************************
ok: [c1_s1_cpe] => {
    "output.stdout_lines": [
        [
            "Interface                  IP-Address      OK? Method Status                ␣
↪    Protocol",
            "FastEthernet0/0            10.0.12.2       YES NVRAM  up                    ␣
↪    up       ",
            "FastEthernet0/1            10.100.12.1     YES NVRAM  up                    ␣
↪    up       ",
            "FastEthernet1/0            unassigned      YES NVRAM  administratively␣
↪down down"
        ]
    ]
}
ok: [c1_s1_co-1] => {
    "output.stdout_lines": [
        [
            "Interface                  IP-Address      OK? Method Status                ␣
↪    Protocol",
            "FastEthernet0/0            10.100.12.2     YES NVRAM  up                    ␣
↪    up       ",
            "FastEthernet0/1            unassigned      YES unset  administratively␣
↪down down    ",
            "FastEthernet1/0            unassigned      YES unset  up                    ␣
↪    up       ",
            "FastEthernet1/1            unassigned      YES unset  up                    ␣
↪    down     ",
            "FastEthernet1/2            unassigned      YES unset  up                    ␣
↪    down     ",
```

(continues on next page)

(continued from previous page)

```
            "FastEthernet1/3           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/4           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/5           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/6           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/7           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/8           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/9           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/10          unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/11          unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/12          unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/13          unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/14          unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/15          unassigned      YES unset  up
↪    down    ",
            "Vlan1                     unassigned      YES NVRAM  administratively␣
↪down down    ",
            "Vlan10                    10.100.200.1    YES NVRAM  up
↪    up"
        ]
    ]
}
ok: [c1_s1_ua-1] => {
    "output.stdout_lines": [
        [
            "Interface                 IP-Address      OK? Method Status
↪    Protocol",
            "FastEthernet0/0           unassigned      YES NVRAM  administratively␣
↪down down    ",
            "FastEthernet0/1           unassigned      YES NVRAM  administratively␣
↪down down    ",
            "FastEthernet1/0           unassigned      YES unset  up
↪    up      ",
            "FastEthernet1/1           unassigned      YES unset  up
↪    up      ",
            "FastEthernet1/2           unassigned      YES unset  up
↪    up      ",
            "FastEthernet1/3           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/4           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/5           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/6           unassigned      YES unset  up
↪    down    ",
            "FastEthernet1/7           unassigned      YES unset  up
↪    down    ",
```

(continues on next page)

(continued from previous page)

```
            "FastEthernet1/8            unassigned      YES unset  up                  ␣
↪    down    ",
            "FastEthernet1/9            unassigned      YES unset  up                  ␣
↪    down    ",
            "FastEthernet1/10           unassigned      YES unset  up                  ␣
↪    down    ",
            "FastEthernet1/11           unassigned      YES unset  up                  ␣
↪    down    ",
            "FastEthernet1/12           unassigned      YES unset  up                  ␣
↪    down    ",
            "FastEthernet1/13           unassigned      YES unset  up                  ␣
↪    down    ",
            "FastEthernet1/14           unassigned      YES unset  up                  ␣
↪    down    ",
            "FastEthernet1/15           unassigned      YES unset  up                  ␣
↪    down    ",
            "Vlan1                      unassigned      YES NVRAM  administratively␣
↪down down    ",
            "Vlan10                     10.100.200.2    YES NVRAM  up                  ␣
↪    up"
        ]
    ]
}

PLAY RECAP␣
↪*********************************************************************************************
c1_s1_co-1                 : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_ua-1                 : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook ios-show-
↪cmd.yml -e "cmd='ip arp'" -l c1_s1

PLAY [IOS show cmd]␣
↪*********************************************************************************************

TASK [IOS show cmd]␣
↪*********************************************************************************************

ok: [c1_s1_cpe]

ok: [c1_s1_co-1]

TASK [debug]␣
↪*********************************************************************************************
ok: [c1_s1_co-1] => {
    "output.stdout_lines": [
        [
            "Protocol  Address          Age (min)  Hardware Addr   Type   Interface",
            "Internet  10.100.12.1            75   c202.5d80.0001  ARPA   ␣
↪FastEthernet0/0",
            "Internet  10.100.12.2             -   c204.5f8c.0000  ARPA   ␣
↪FastEthernet0/0",
```

(continues on next page)

(continued from previous page)

```
            "Internet  10.100.200.1            -   c204.5f8c.0000  ARPA   Vlan10",
            "Internet  10.100.200.2           75   c206.1b68.0000  ARPA   Vlan10"
        ]
    ]
}
ok: [c1_s1_cpe] => {
    "output.stdout_lines": [
        [
            "Protocol  Address          Age (min)  Hardware Addr   Type   Interface",
            "Internet  10.0.12.1              75   c201.375c.0001  ARPA   ␣
↪FastEthernet0/0",
            "Internet  10.0.12.2               -   c202.5d80.0000  ARPA   ␣
↪FastEthernet0/0",
            "Internet  10.100.12.1             -   c202.5d80.0001  ARPA   ␣
↪FastEthernet0/1",
            "Internet  10.100.12.2            75   c204.5f8c.0000  ARPA   ␣
↪FastEthernet0/1"
        ]
    ]
}

PLAY RECAP␣
↪*********************************************************************************************
c1_s1_co-1                 : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

**show interfaces**

This playbook is using the ansible network engine role and/with a parser, which means that the standard output is being send to an external script to crop that output and give back the results to Ansible to show it.

You can still get the same info in a simpler way, the interesting part here is to show the power of roles, parses, and scripts, in order to process the regular output.

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook ne-
↪showintf.yml -l c1_s1

PLAY [GENERATE A REPORT]␣
↪*********************************************************************************************

TASK [CAPTURE SHOW IP INTERFACE]␣
↪*********************************************************************************************

ok: [c1_s1_cpe]

ok: [c1_s1_co-1]

TASK [PARSE THE RAW OUTPUT]␣
↪*********************************************************************************************

ok: [c1_s1_cpe]
ok: [c1_s1_co-1]

TASK [Display the data]␣
↪*********************************************************************************************
```

(continues on next page)

(continued from previous page)


```
ok: [c1_s1_cpe] => {
    "interface_facts": {
        "FastEthernet0/0": {
            "config": {
                "description": "to_r1",
                "mtu": "1500",
                "name": "FastEthernet0/0",
                "type": null
            }
        },
        "FastEthernet0/1": {
            "config": {
                "description": "to_inside",
                "mtu": "1500",
                "name": "FastEthernet0/1",
                "type": "AmdFE"
            }
        }
    }
}
ok: [c1_s1_co-1] => {
    "interface_facts": {
        "FastEthernet0/0": {
            "config": {
                "description": "to_inet",
                "mtu": "1500",
                "name": "FastEthernet0/0",
                "type": null
            }
        },
        "FastEthernet1/0": {
            "config": {
                "description": null,
                "mtu": "1500",
                "name": "FastEthernet1/0",
                "type": null
            }
        },
        "FastEthernet1/1": {
            "config": {
                "description": null,
                "mtu": "1500",
                "name": "FastEthernet1/1",
                "type": null
            }
        },
        "FastEthernet1/10": {
            "config": {
                "description": null,
                "mtu": "1500",
                "name": "FastEthernet1/10",
                "type": null
            }
        },
        "FastEthernet1/11": {
            "config": {
                "description": null,
```


(continues on next page)

(continued from previous page)


```
            "mtu": "1500",
            "name": "FastEthernet1/11",
            "type": null
        }
    },
    "FastEthernet1/12": {
        "config": {
            "description": null,
            "mtu": "1500",
            "name": "FastEthernet1/12",
            "type": null
        }
    },
    "FastEthernet1/13": {
        "config": {
            "description": null,
            "mtu": "1500",
            "name": "FastEthernet1/13",
            "type": null
        }
    },
    "FastEthernet1/14": {
        "config": {
            "description": null,
            "mtu": "1500",
            "name": "FastEthernet1/14",
            "type": null
        }
    },
    "FastEthernet1/15": {
        "config": {
            "description": null,
            "mtu": "1500",
            "name": "FastEthernet1/15",
            "type": "EtherSVI"
        }
    },
    "FastEthernet1/2": {
        "config": {
            "description": null,
            "mtu": "1500",
            "name": "FastEthernet1/2",
            "type": null
        }
    },
    "FastEthernet1/3": {
        "config": {
            "description": null,
            "mtu": "1500",
            "name": "FastEthernet1/3",
            "type": null
        }
    },
    "FastEthernet1/4": {
        "config": {
            "description": null,
            "mtu": "1500",
```


(continues on next page)

(continued from previous page)

```
                "name": "FastEthernet1/4",
                "type": null
            }
        },
        "FastEthernet1/5": {
            "config": {
                "description": null,
                "mtu": "1500",
                "name": "FastEthernet1/5",
                "type": null
            }
        },
        "FastEthernet1/6": {
            "config": {
                "description": null,
                "mtu": "1500",
                "name": "FastEthernet1/6",
                "type": null
            }
        },
        "FastEthernet1/7": {
            "config": {
                "description": null,
                "mtu": "1500",
                "name": "FastEthernet1/7",
                "type": null
            }
        },
        "FastEthernet1/8": {
            "config": {
                "description": null,
                "mtu": "1500",
                "name": "FastEthernet1/8",
                "type": null
            }
        },
        "FastEthernet1/9": {
            "config": {
                "description": null,
                "mtu": "1500",
                "name": "FastEthernet1/9",
                "type": null
            }
        },
        "Vlan10": {
            "config": {
                "description": "LAN",
                "mtu": "1500",
                "name": "Vlan10",
                "type": "EtherSVI"
            }
        }
    }
}

PLAY RECAP␣
↪*********************************************************************************************************
```

(continues on next page)

(continued from previous page)

```
c1_s1_co-1                 : ok=3    changed=0    unreachable=0    failed=0
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=3    changed=0    unreachable=0    failed=0
↪skipped=0    rescued=0    ignored=0
```

** Another example of parsers to show ip interface brief**

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook ne-
↪showipintf.yml -l c1_s1

PLAY [GENERATE A REPORT]
↪*********************************************************************************************

TASK [CAPTURE SHOW IP INTERFACE]
↪*****************************************************************************************

ok: [c1_s1_cpe]

ok: [c1_s1_co-1]


TASK [PARSE THE RAW OUTPUT]
↪*********************************************************************************************

ok: [c1_s1_co-1]
ok: [c1_s1_ua-1]
ok: [c1_s1_cpe]

TASK [DISPLAY THE DATA]
↪*********************************************************************************************
ok: [c1_s1_cpe] => {
    "ip_interface_facts": [
        {
            "FastEthernet0/0": {
                "data": {
                    "admin_state": "up",
                    "ip": "10.0.12.2",
                    "name": "FastEthernet0/0",
                    "protocol_state": "up"
                }
            }
        },
        {
            "FastEthernet0/1": {
                "data": {
                    "admin_state": "up",
                    "ip": "10.100.12.1",
                    "name": "FastEthernet0/1",
                    "protocol_state": "up"
                }
            }
        }
    ]
}
ok: [c1_s1_co-1] => {
    "ip_interface_facts": [
        {
```

(continues on next page)

(continued from previous page)

```
            "FastEthernet0/0": {
                "data": {
                    "admin_state": "up",
                    "ip": "10.100.12.2",
                    "name": "FastEthernet0/0",
                    "protocol_state": "up"
                }
            }
        },
        {
            "Vlan10": {
                "data": {
                    "admin_state": "up",
                    "ip": "10.100.200.1",
                    "name": "Vlan10",
                    "protocol_state": "up"
                }
            }
        }
    ]
}

PLAY RECAP␣
↪*********************************************************************************************************
c1_s1_co-1                 : ok=3    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=3    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

**this example send the output to a python script which proceses the data and returns a dictornary to Ansible in order to format the output**

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook ne-show-
↪ver.yml -l c1_s1

PLAY [Show Cisco HW, SN, and SW version]␣
↪*********************************************************************************

TASK [Show version]␣
↪*********************************************************************************************************

ok: [c1_s1_cpe]

ok: [c1_s1_co-1]

ok: [c1_s1_ua-1]

TASK [PARSE THE RAW OUTPUT]␣
↪*********************************************************************************************************

ok: [c1_s1_ua-1]
ok: [c1_s1_co-1]
ok: [c1_s1_cpe]

TASK [execute python script]␣
↪*********************************************************************************************************
```

(continues on next page)

(continued from previous page)

```
changed: [c1_s1_co-1 -> localhost]
changed: [c1_s1_ua-1 -> localhost]
changed: [c1_s1_cpe -> localhost]

TASK [debug]␣
↪*********************************************************************************************
ok: [c1_s1_co-1] => {
    "output.stdout_lines": [
        "Hostname: c1_s1_co-1",
        "Serial_Number: FTX0945W0MY",
        "Software_Release: fc3",
        "Hardware_Version: 3725",
        "Software_Version: 12.4(15)T13",
        "Software_Image: C3725-ADVENTERPRISEK9-M"
    ]
}
ok: [c1_s1_cpe] => {
    "output.stdout_lines": [
        "Hostname: c1_s1_cpe",
        "Serial_Number: FTX0945W0MY",
        "Software_Release: fc3",
        "Hardware_Version: 3725",
        "Software_Version: 12.4(15)T13",
        "Software_Image: C3725-ADVENTERPRISEK9-M"
    ]
}
ok: [c1_s1_ua-1] => {
    "output.stdout_lines": [
        "Hostname: c1_s1_ua-1",
        "Serial_Number: FTX0945W0MY",
        "Software_Release: fc3",
        "Hardware_Version: 3725",
        "Software_Version: 12.4(15)T13",
        "Software_Image: C3725-ADVENTERPRISEK9-M"
    ]
}

PLAY RECAP␣
↪*********************************************************************************************
c1_s1_co-1                 : ok=4    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=4    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_ua-1                 : ok=4    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

**example on how to set up snmp**

In this case, the configuration was applied to two devices, because the 3rd one already had it. Look for the word "changed"

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook set-snmp.
↪yml -l c1_s1

PLAY [Set SNMP]␣
↪*********************************************************************************************
```

(continues on next page)

(continued from previous page)

```
TASK [Configure SNMP comminities on devices]␣
↪*********************************************************************************

ok: [c1_s1_cpe]

changed: [c1_s1_co-1]

changed: [c1_s1_ua-1]

PLAY RECAP␣
↪*********************************************************************************************************
c1_s1_co-1                 : ok=1    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=1    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_ua-1                 : ok=1    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0

adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook ios-show-
↪cmd.yml -e "cmd='run | inc snmp'" -l c1_s1

PLAY [IOS show cmd]␣
↪*********************************************************************************************************

TASK [IOS show cmd]␣
↪*********************************************************************************************************

ok: [c1_s1_cpe]

ok: [c1_s1_co-1]

ok: [c1_s1_ua-1]

TASK [debug]␣
↪*********************************************************************************************************
ok: [c1_s1_cpe] => {
    "output.stdout_lines": [
        [
            "snmp-server community snmpCommunity RW",
            "snmp-server community read_only RO",
            "snmp-server community read_write RW"
        ]
    ]
}
ok: [c1_s1_co-1] => {
    "output.stdout_lines": [
        [
            "snmp-server community snmpCommunity RW",
            "snmp-server community read_only RO",
            "snmp-server community read_write RW"
        ]
    ]
}
ok: [c1_s1_ua-1] => {
    "output.stdout_lines": [
        [
            "snmp-server community snmpCommunity RW",
```

(continues on next page)

(continued from previous page)

```
            "snmp-server community read_only RO",
            "snmp-server community read_write RW"
        ]
    ]
}

PLAY RECAP␣
↪*********************************************************************************************************
c1_s1_co-1                 : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_ua-1                 : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

**another case of show arp**

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook show-arp.
↪yml -l c1_s1

PLAY [Show IP ARP]␣
↪*********************************************************************************************************

TASK [Show IP ARP]␣
↪*********************************************************************************************************

ok: [c1_s1_cpe]

ok: [c1_s1_co-1]

ok: [c1_s1_ua-1]

TASK [debug]␣
↪*********************************************************************************************************
ok: [c1_s1_co-1] => {
    "list_of_ip_arp.stdout_lines": [
        [
            "Protocol  Address          Age (min)  Hardware Addr   Type   Interface",
            "Internet  10.100.12.1            78   c202.5d80.0001  ARPA   ␣
↪FastEthernet0/0",
            "Internet  10.100.12.2             -   c204.5f8c.0000  ARPA   ␣
↪FastEthernet0/0",
            "Internet  10.100.200.1            -   c204.5f8c.0000  ARPA   Vlan10",
            "Internet  10.100.200.2           78   c206.1b68.0000  ARPA   Vlan10"
        ]
    ]
}
ok: [c1_s1_ua-1] => {
    "list_of_ip_arp.stdout_lines": [
        [
            "Protocol  Address          Age (min)  Hardware Addr   Type   Interface",
            "Internet  10.100.200.1           78   c204.5f8c.0000  ARPA   Vlan10",
            "Internet  10.100.200.2            -   c206.1b68.0000  ARPA   Vlan10"
        ]
    ]
}
ok: [c1_s1_cpe] => {
```

(continues on next page)

(continued from previous page)

```
    "list_of_ip_arp.stdout_lines": [
        [
            "Protocol  Address          Age (min)  Hardware Addr   Type   Interface",
            "Internet  10.0.12.1              78   c201.375c.0001  ARPA  ␣
↪FastEthernet0/0",
            "Internet  10.0.12.2               -   c202.5d80.0000  ARPA  ␣
↪FastEthernet0/0",
            "Internet  10.100.12.1             -   c202.5d80.0001  ARPA  ␣
↪FastEthernet0/1",
            "Internet  10.100.12.2            78   c204.5f8c.0000  ARPA  ␣
↪FastEthernet0/1"
        ]
    ]
}

PLAY RECAP␣
↪*********************************************************************************************************
c1_s1_co-1                 : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_ua-1                 : ok=2    changed=0    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

**getting better, this one checks if an ACL is already there, and if not it will apply it**

```
adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook check-acl.
↪yml -l c1_s1 --check

PLAY [Check or create exact ACL order]␣
↪*********************************************************************************************

TASK [Check or create exact ACL order]␣
↪*********************************************************************************************

changed: [c1_s1_cpe]

changed: [c1_s1_co-1]

changed: [c1_s1_ua-1]

PLAY RECAP␣
↪*********************************************************************************************************
c1_s1_co-1                 : ok=1    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=1    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_ua-1                 : ok=1    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0

adrian@adrian-VirtualBox:~/netor/netor/ansible/playbooks$ ansible-playbook check-acl.
↪yml -l c1_s1 --check --diff

PLAY [Check or create exact ACL order]␣
↪*********************************************************************************************

TASK [Check or create exact ACL order]␣
↪*********************************************************************************************
```

(continues on next page)

(continued from previous page)

```
changed: [c1_s1_cpe]

changed: [c1_s1_co-1]

changed: [c1_s1_ua-1]

PLAY RECAP␣
↪*********************************************************************************************************
c1_s1_co-1                 : ok=1    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_cpe                  : ok=1    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
c1_s1_ua-1                 : ok=1    changed=1    unreachable=0    failed=0    ␣
↪skipped=0    rescued=0    ignored=0
```

## 5.2 Salt examples

From easy to hard

**basic to test the connection between Salt and the devices**

```
adrian@adrian-VirtualBox:~$ sudo salt 'c1_s1*' test.ping
c1_s1_ua-1:
    True
c1_s1_co-1:
    True
c1_s1_cpe:
    True
```

**you can also add the ‘‘-l debug‘‘ flag**

```
adrian@adrian-VirtualBox:~$ sudo salt 'c1_s1*' test.ping -l debug
[DEBUG   ] Reading configuration from /etc/salt/master
[DEBUG   ] Using cached minion ID from /etc/salt/minion_id: adrian-VirtualBox
[DEBUG   ] Missing configuration file: /home/adrian/.saltrc
[DEBUG   ] Configuration file path: /etc/salt/master
[WARNING ] Insecure logging configuration detected! Sensitive data may be logged.
[DEBUG   ] Reading configuration from /etc/salt/master
[DEBUG   ] Using cached minion ID from /etc/salt/minion_id: adrian-VirtualBox
[DEBUG   ] Missing configuration file: /home/adrian/.saltrc
[DEBUG   ] MasterEvent PUB socket URI: /var/run/salt/master/master_event_pub.ipc
[DEBUG   ] MasterEvent PULL socket URI: /var/run/salt/master/master_event_pull.ipc
[DEBUG   ] Initializing new AsyncZeroMQReqChannel for ('/home/adrian/netor-master/
↪netor/salt/config/pki/master', 'adrian-VirtualBox_master', 'tcp://127.0.0.1:4506',
↪'clear')
[DEBUG   ] Connecting the Minion to the Master URI (for the return server): tcp://127.
↪0.0.1:4506
[DEBUG   ] Trying to connect to: tcp://127.0.0.1:4506
[DEBUG   ] Closing AsyncZeroMQReqChannel instance
[DEBUG   ] LazyLoaded local_cache.get_load
[DEBUG   ] Reading minion list from /var/cache/salt/master/jobs/ba/
↪6ceb1709725e52888fafec43611acca92cb7287fe14f0aab323f7711bbc3f0/.minions.p
[DEBUG   ] get_iter_returns for jid 20191116123204208193 sent to {'c1_s1_cpe', 'c1_s1_
↪co-1', 'c1_s1_ua-1'} will timeout at 12:32:09.226416
```

(continues on next page)

(continued from previous page)

```
[DEBUG   ] jid 20191116123204208193 return from c1_s1_ua-1
[DEBUG   ] return event: {'c1_s1_ua-1': {'ret': True, 'retcode': 0, 'jid':
↪'20191116123204208193'}}
[DEBUG   ] LazyLoaded nested.output
c1_s1_ua-1:
    True
[DEBUG   ] jid 20191116123204208193 return from c1_s1_cpe
[DEBUG   ] return event: {'c1_s1_cpe': {'ret': True, 'retcode': 0, 'jid':
↪'20191116123204208193'}}
[DEBUG   ] LazyLoaded nested.output
c1_s1_cpe:
    True
[DEBUG   ] jid 20191116123204208193 return from c1_s1_co-1
[DEBUG   ] return event: {'c1_s1_co-1': {'ret': True, 'retcode': 0, 'jid':
↪'20191116123204208193'}}
[DEBUG   ] LazyLoaded nested.output
c1_s1_co-1:
    True
[DEBUG   ] jid 20191116123204208193 found all minions {'c1_s1_cpe', 'c1_s1_ua-1', 'c1_
↪s1_co-1'}
[DEBUG   ] Closing IPCMessageSubscriber instance
adrian@adrian-VirtualBox:~$
```

**this is i think the coolest and easiest function of Salt**

The **net.find** module allows you to search in 3 seconds information gathered by mining. Lets look for IP address, MACs, interface descriptions, vlan, etc. configured on the devices.

```
adrian@adrian-VirtualBox:~/netor/netor/salt$ sudo salt-run net.find 10.0.0.0/8␣
↪best=False
Details for all interfaces that include network 10.0.0.0/8

    ----------------------------------------------------------------------------------
↪----------------------------------------
    |   Device   |    Interface    | Interface Description |   IP Addresses   |␣
↪Enabled |  UP  |    MAC Address    | Speed [Mbps] |
    ----------------------------------------------------------------------------------
↪----------------------------------------
    | c1_s1_co-1 | FastEthernet0/0 |        to_inet        |  10.100.12.2/24 |   True␣
↪ | True | C2:04:5F:8C:00:00 |      10      |
    ----------------------------------------------------------------------------------
↪----------------------------------------
    | c1_s1_co-1 |      Vlan10     |          LAN          | 10.100.200.1/24 |   True␣
↪ | True | C2:04:5F:8C:00:00 |     100      |
    ----------------------------------------------------------------------------------
↪----------------------------------------
    | c1_s1_cpe  | FastEthernet0/0 |         to_r1         |   10.0.12.2/24  |   True␣
↪ | True | C2:02:5D:80:00:00 |      10      |
    ----------------------------------------------------------------------------------
↪----------------------------------------
    | c1_s1_cpe  | FastEthernet0/1 |       to_inside       |  10.100.12.1/24 |   True␣
↪ | True | C2:02:5D:80:00:01 |      10      |
    ----------------------------------------------------------------------------------
↪----------------------------------------
    | c1_s1_ua-1 |      Vlan10     |         user1         | 10.100.200.2/24 |   True␣
↪ | True | C2:06:1B:68:00:00 |     100      |
    ----------------------------------------------------------------------------------
↪----------------------------------------
```

(continues on next page)

(continued from previous page)

```
    | c2_s1_co-1 | FastEthernet0/0 |        to_inet        |  10.101.23.2/24 |   True␣
↪ | True | C2:05:48:3C:00:00 |      10      |
    ------------------------------------------------------------------------------------
↪----------------------------------------
    | c2_s1_co-1 |      Vlan10     |          LAN          | 10.101.201.1/24 |   True␣
↪ | True | C2:05:48:3C:00:00 |     100      |
    ------------------------------------------------------------------------------------
↪----------------------------------------
    | c2_s1_cpe  | FastEthernet0/0 |         to_r1         |   10.0.13.2/24  |   True␣
↪ | True | C2:03:29:20:00:00 |      10      |
    ------------------------------------------------------------------------------------
↪----------------------------------------
    | c2_s1_cpe  | FastEthernet0/1 |       to_inside       |  10.101.23.1/24 |   True␣
↪ | True | C2:03:29:20:00:01 |      10      |
    ------------------------------------------------------------------------------------
↪----------------------------------------
    | c2_s1_ua-1 |      Vlan10     |         user1         | 10.101.201.2/24 |   True␣
↪ | True | C2:07:61:70:00:00 |     100      |
    ------------------------------------------------------------------------------------
↪----------------------------------------
None
```

```
adrian@adrian-VirtualBox:~/netor/netor/salt$ sudo salt-run net.find Vlan10
Pattern "Vlan10" found in the description of the following interfaces
Details for interface Vlan10

    ------------------------------------------------------------------------------------
↪-------------------------------------
    |   Device   | Interface | Interface Description |   IP Addresses  | Enabled | ␣
↪UP  |    MAC Address    | Speed [Mbps] |
    ------------------------------------------------------------------------------------
↪-------------------------------------
    | c1_s1_ua-1 |   Vlan10  |         user1         | 10.100.200.2/24 |   True  |␣
↪True | C2:06:1B:68:00:00 |     100      |
    ------------------------------------------------------------------------------------
↪-------------------------------------
    | c2_s1_ua-1 |   Vlan10  |         user1         | 10.101.201.2/24 |   True  |␣
↪True | C2:07:61:70:00:00 |     100      |
    ------------------------------------------------------------------------------------
↪-------------------------------------
Details for all interfaces on device Vlan10
Pattern "Vlan10" found in one of the following LLDP details
LLDP Neighbors for interface Vlan10
LLDP Neighbors for all interfaces on device Vlan10
MAC Address(es) on device Vlan10
MAC Address(es) on interface Vlan10
ARP Entries on device Vlan10
ARP Entries on interface Vlan10

    ---------------------------------------------------------------------
    |  Age  |   Device   | Interface |       IP      |        MAC        |
    ---------------------------------------------------------------------
    | 108.0 | c1_s1_ua-1 |   Vlan10  | 10.100.200.1 | C2:04:5F:8C:00:00 |
    ---------------------------------------------------------------------
    |  0.0  | c1_s1_ua-1 |   Vlan10  | 10.100.200.2 | C2:06:1B:68:00:00 |
    ---------------------------------------------------------------------
```

(continues on next page)

(continued from previous page)

```
    | 108.0 | c2_s1_ua-1 |   Vlan10   | 10.101.201.1 | C2:05:48:3C:00:00 |
    ------------------------------------------------------------------------
    |  0.0  | c2_s1_ua-1 |   Vlan10   | 10.101.201.2 | C2:07:61:70:00:00 |
    ------------------------------------------------------------------------
```

```
adrian@adrian-VirtualBox:~/netor/netor/salt$ sudo salt-run net.find to_inside
Pattern "to_inside" found in the description of the following interfaces

    ------------------------------------------------------------------------------------
↪----------------------------------------
    |   Device   |    Interface    | Interface Description |  IP Addresses  | Enabled␣
↪|  UP   |    MAC Address    | Speed [Mbps] |
    ------------------------------------------------------------------------------------
↪----------------------------------------
    | c1_s1_cpe | FastEthernet0/1 |       to_inside       | 10.100.12.1/24 |   True ␣
↪| True | C2:02:5D:80:00:01 |      10      |
    ------------------------------------------------------------------------------------
↪----------------------------------------
Details for interface to_inside
Details for all interfaces on device to_inside
Pattern "to_inside" found in one of the following LLDP details
LLDP Neighbors for interface to_inside
LLDP Neighbors for all interfaces on device to_inside
MAC Address(es) on device to_inside
MAC Address(es) on interface to_inside
ARP Entries on device to_inside
ARP Entries on interface to_inside
None
```

```
adrian@adrian-VirtualBox:~/netor/netor/salt$ sudo salt-run net.find 10.100.12.1
Details for all interfaces that include network 10.100.12.1/32 - only best match␣
↪returned

    ------------------------------------------------------------------------------------
↪----------------------------------------
    |   Device   |    Interface    | Interface Description |  IP Addresses  | Enabled␣
↪|  UP   |    MAC Address    | Speed [Mbps] |
    ------------------------------------------------------------------------------------
↪----------------------------------------
    | c1_s1_cpe | FastEthernet0/1 |       to_inside       | 10.100.12.1/24 |   True ␣
↪| True | C2:02:5D:80:00:01 |      10      |
    ------------------------------------------------------------------------------------
↪----------------------------------------
ARP Entries for IP 10.100.12.1

    ------------------------------------------------------------------------
    | Age |   Device   |    Interface    |      IP      |        MAC        |
    ------------------------------------------------------------------------
    | 0.0 | c1_s1_cpe | FastEthernet0/1 | 10.100.12.1 | C2:02:5D:80:00:01 |
    ------------------------------------------------------------------------
IP Address 10.100.12.1 is set for interface FastEthernet0/1, on c1_s1_cpe

    ------------------------------------------------------------------------------------
↪----------------------------------------
    |   Device   |    Interface    | Interface Description |  IP Addresses  | Enabled␣
↪|  UP   |    MAC Address    | Speed [Mbps] |
```

(continues on next page)

(continued from previous page)

```
    ------------------------------------------------------------------------------------
↪---------------------------------------
    | c1_s1_cpe | FastEthernet0/1 |       to_inside       | 10.100.12.1/24 |   True ␣
↪| True | C2:02:5D:80:00:01 |      10      |
    ------------------------------------------------------------------------------------
↪---------------------------------------
LLDP Neighbors for interface FastEthernet0/1 on device c1_s1_cpe
None
```

```
adrian@adrian-VirtualBox:~/netor/netor/salt$ sudo salt-run net.find C2:02:5D:80:00:01
MAC Address(es)
ARP Entries for MAC C2:02:5D:80:00:01

    ------------------------------------------------------------------------
    |  Age  |   Device   |    Interface    |      IP     |        MAC        |
    ------------------------------------------------------------------------
    | 114.0 | c1_s1_co-1 | FastEthernet0/0 | 10.100.12.1 | C2:02:5D:80:00:01 |
    ------------------------------------------------------------------------
LLDP Neighbors for all interfaces having Chassis ID C2:02:5D:80:00:01
Interface FastEthernet0/1 on c1_s1_cpe has the physical address (C2:02:5D:80:00:01)

    ------------------------------------------------------------------------------------
↪---------------------------------------
    |   Device   |    Interface    | Interface Description |  IP Addresses  | Enabled␣
↪|  UP  |    MAC Address    | Speed [Mbps] |
    ------------------------------------------------------------------------------------
↪---------------------------------------
    | c1_s1_cpe | FastEthernet0/1 |       to_inside       | 10.100.12.1/24 |   True ␣
↪| True | C2:02:5D:80:00:01 |      10      |
    ------------------------------------------------------------------------------------
↪---------------------------------------
LLDP Neighbors for interface FastEthernet0/1 on device c1_s1_cpe
None
```

**States, great concept!**

It is getting better…

Salt define a **sate** in a file in which you can define attributes (like ntp in this example), and later you can apply that state/attribute to any OS. Yes it will figure out what commands to execute depending on the OS.

Read about this state ntp.sls file at the `netor/salt/config/pillar/states` folder.

```
adrian@adrian-VirtualBox:~/netor/netor/salt$ sudo salt 'c1_s1_cpe' state.apply ntp
c1_s1_cpe:
----------
          ID: netntp
    Function: netntp.managed
      Result: True
     Comment: Device configured properly.
     Started: 23:44:39.097859
    Duration: 1629.019 ms
     Changes:

Summary for c1_s1_cpe
------------
Succeeded: 1
```

(continues on next page)

(continued from previous page)

```
Failed:    0
------------
Total states run:     1
Total run time:   1.629 s
adrian@adrian-VirtualBox:~/netor/netor/salt$


adrian@adrian-VirtualBox:~/netor/netor/salt$ more ./config/pillar/states/ntp.sls
netntp:
  netntp.managed:
    - servers:
      - 10.0.0.2
adrian@adrian-VirtualBox:~/netor/netor/salt$
```

**this is how you can view the event bus**

You will see what happens when you apply the **state**

```
adrian@adrian-VirtualBox:~/netor/netor/salt$ sudo salt-run state.event pretty=True
20191115234741088036        {
    "_stamp": "2019-11-15T22:47:41.088306",
    "minions": [
        "c1_s1_cpe"
    ]
}
salt/job/20191115234741088036/new   {
    "_stamp": "2019-11-15T22:47:41.088725",
    "arg": [
        "ntp"
    ],
    "fun": "state.apply",
    "jid": "20191115234741088036",
    "minions": [
        "c1_s1_cpe"
    ],
    "missing": [],
    "tgt": "c1_s1_cpe",
    "tgt_type": "glob",
    "user": "sudo_adrian"
}
minion/refresh/c1_s1_cpe    {
    "Minion data cache refresh": "c1_s1_cpe",
    "_stamp": "2019-11-15T22:47:41.300837"
}
salt/job/20191115234741088036/ret/c1_s1_cpe {
    "_stamp": "2019-11-15T22:47:43.462567",
    "cmd": "_return",
    "fun": "state.apply",
    "fun_args": [
        "ntp"
    ],
    "id": "c1_s1_cpe",
    "jid": "20191115234741088036",
    "out": "highstate",
    "retcode": 0,
    "return": {
        "netntp_-netntp_-netntp_-managed": {
```

(continues on next page)

(continued from previous page)

```
            "__id__": "netntp",
            "__run_num__": 0,
            "__sls__": "ntp",
            "changes": {},
            "comment": "Device configured properly.",
            "duration": 2026.341,
            "name": "netntp",
            "result": true,
            "start_time": "23:47:41.424906"
        }
    },
    "success": true
}
```

**how to use the online help of the commands**

In this case the mine function/module

```
adrian@lmint2:~$ sudo salt-run mine
mine.get:

        Gathers the data from the specified minions' mine, pass in the target,
        function to look up and the target type

        CLI Example:

            salt-run mine.get '*' network.interfaces

mine.update:

        New in version 2017.7.0

        Update the mine data on a certain group of minions.

        tgt
            Which minions to target for the execution.

        tgt_type: ``glob``
            The type of ``tgt``.

        clear: ``False``
            Boolean flag specifying whether updating will clear the existing
            mines, or will update. Default: ``False`` (update).

        mine_functions
            Update the mine data on certain functions only.
            This feature can be used when updating the mine for functions
            that require refresh at different intervals than the rest of
            the functions specified under ``mine_functions`` in the
            minion/master config or pillar.

        CLI Example:

            salt-run mine.update '*'
            salt-run mine.update 'juniper-edges' tgt_type='nodegroup'

... continue
```

**wait you can do a simulation with the "test=True" option**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' state.apply sla test=True
c1_s1_cpe:
----------
          ID: rpmprobes
    Function: probes.managed
      Result: None
     Comment: Testing mode: configuration was not changed!
     Started: 10:37:24.816077
    Duration: 1648.158 ms
     Changes:
              ----------
              added:
                  ----------
                  probe_name1:
                      ----------
                      probe1_test1:
                          ----------
                          probe_type:
                              icmp-ping
                          target:
                              10.0.12.1
                      probe1_test2:
                          ----------
                          probe_count:
                              3
                          probe_type:
                              udp-ping
                          source:
                              10.100.12.1
                          target:
                              10.0.12.1
                          test_interval:
                              5
              removed:
                  None
              updated:
                  None

Summary for c1_s1_cpe
------------
Succeeded: 1 (unchanged=1, changed=1)
Failed:    0
------------
Total states run:     1
Total run time:   1.648 s
```

**check a running configuration**

This command will take 3 second since you can have a proxy minion with a session already established with the device

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.config source='running'
c1_s1_cpe:
    ----------
    comment:
    out:
        ----------
```

(continues on next page)

(continued from previous page)

```
        candidate:
        running:
            Building configuration...

            Current configuration : 2202 bytes
            !
            version 12.4
            no service pad
            service tcp-keepalives-in
            service tcp-keepalives-out
            service timestamps debug datetime msec localtime show-timezone
            service timestamps log datetime msec localtime show-timezone
            service password-encryption
            !
            hostname r2
            !
            boot-start-marker
            boot-end-marker
            !
            logging buffered 32000
            no logging console
            enable secret 5 $1$QAh2$FiUShFDsaikloAgWmKsW1.
            !
            aaa new-model
            !
            !
            aaa authentication login default local-case
            aaa authorization exec default local
            !
            !
            aaa session-id common
            memory-size iomem 5
            no ip source-route
            ip options drop
            ip cef
            !
            !
            ip dhcp bootp ignore
            !
            !
            no ip domain lookup
            ip domain name quadrant.edu
            !
            multilink bundle-name authenticated
... continue
```

**of course you can add a simple 'grep'**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.config source='running' | grep snmp
            snmp-server community snmpCommunity RW
            snmp-server community read_only RO
            snmp-server community read_write RW
```

** do a simple ping from several devices to check for problems**

You could try this for ping from several countries/sites to 1 server/service inside/outside of the network.

```
adrian@adrian-VirtualBox:~$ sudo salt 'c1_s1_*' network.ping 10.0.12.2
c1_s1_ua-1:
    PING 10.0.12.2 (10.0.12.2) 56(84) bytes of data.
    64 bytes from 10.0.12.2: icmp_seq=1 ttl=253 time=31.9 ms
    64 bytes from 10.0.12.2: icmp_seq=2 ttl=253 time=324 ms
    64 bytes from 10.0.12.2: icmp_seq=3 ttl=253 time=21.4 ms
    64 bytes from 10.0.12.2: icmp_seq=4 ttl=253 time=103 ms

    --- 10.0.12.2 ping statistics ---
    4 packets transmitted, 4 received, 0% packet loss, time 3003ms
    rtt min/avg/max/mdev = 21.461/120.435/324.668/122.081 ms
c1_s1_cpe:
    PING 10.0.12.2 (10.0.12.2) 56(84) bytes of data.
    64 bytes from 10.0.12.2: icmp_seq=1 ttl=253 time=41.7 ms
    64 bytes from 10.0.12.2: icmp_seq=2 ttl=253 time=344 ms
    64 bytes from 10.0.12.2: icmp_seq=3 ttl=253 time=52.1 ms
    64 bytes from 10.0.12.2: icmp_seq=4 ttl=253 time=124 ms

    --- 10.0.12.2 ping statistics ---
    4 packets transmitted, 4 received, 0% packet loss, time 3003ms
    rtt min/avg/max/mdev = 41.770/140.752/344.745/121.997 ms
c1_s1_co-1:
    PING 10.0.12.2 (10.0.12.2) 56(84) bytes of data.
    64 bytes from 10.0.12.2: icmp_seq=1 ttl=253 time=44.9 ms
    64 bytes from 10.0.12.2: icmp_seq=2 ttl=253 time=359 ms
    64 bytes from 10.0.12.2: icmp_seq=3 ttl=253 time=66.6 ms
    64 bytes from 10.0.12.2: icmp_seq=4 ttl=253 time=148 ms

    --- 10.0.12.2 ping statistics ---
    4 packets transmitted, 4 received, 0% packet loss, time 3005ms
    rtt min/avg/max/mdev = 44.999/155.004/359.790/124.385 ms
```

**if the have a route to a destination**

```
adrian@lmint2:~$ sudo salt '*' route.show 192.168.201.3
c1_s1_co-1:
    ----------
    comment:
    out:
        ----------
        192.168.201.3:
    result:
        True
c2_s1_ua-1:
    ----------
    comment:
    out:
        ----------
        192.168.201.3:
    result:
        True
c1_s1_cpe:
    ----------
    comment:
    out:
        ----------
        192.168.201.3:
```

(continues on next page)

(continued from previous page)

```
    result:
        True
c2_s1_cpe:
    ----------
    comment:
    out:
        ----------
        192.168.201.3:
    result:
        True
```

**a simple ping with True or False if it was successful**

```
adrian@lmint2:~$ sudo salt 'c1_s1*' net.ping 192.168.201.3
c1_s1_ua-1:
    ----------
    comment:
    out:
        ----------
    result:
        True
c1_s1_cpe:
    ----------
    comment:
    out:
        ----------
    result:
        True
```

**a traceroute showing the latency**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.traceroute 192.168.201.3
c1_s1_cpe:
    ----------
    comment:
    out:
        ----------
        success:
            ----------
            0:
                ----------
                probes:
                    ----------
                    1:
                        ----------
                        host_name:
                            10.0.12.1
                        ip_address:
                            10.0.12.1
                        rtt:
                            208.0
                    2:
                        ----------
                        host_name:
                            10.0.12.1
                        ip_address:
                            10.0.12.1
```

(continues on next page)

(continued from previous page)

```
                    rtt:
                        32.0
                3:
                    ----------
                    host_name:
                        10.0.12.1
                    ip_address:
                        10.0.12.1
                    rtt:
                        24.0
        1:
            ----------
            probes:
                ----------
                1:
                    ----------
                    host_name:
                        10.0.12.1
                    ip_address:
                        10.0.12.1
                    rtt:
                        28.0
                2:
                    ----------
                    host_name:
                        10.0.12.1
                    ip_address:
                        10.0.12.1
                    rtt:
                        32.0
                3:
                    ----------
                    host_name:
                        10.0.12.1
                    ip_address:
                        10.0.12.1
                    rtt:
                        32.0
        2:
            ----------
            probes:
                ----------
                1:
                    ----------
                    host_name:
                        10.0.0.1
                    ip_address:
                        10.0.0.1
                    rtt:
                        36.0
                2:
                    ----------
                    host_name:
                        10.0.0.1
                    ip_address:
                        10.0.0.1
                    rtt:
```

(continues on next page)

(continued from previous page)

```
                            40.0
                    3:
                        ----------
                        host_name:
                            10.0.0.1
                        ip_address:
                            10.0.0.1
                        rtt:
                            36.0
            3:
                ----------
                probes:
                    ----------
                    1:
                        ----------
                        host_name:
                            192.168.201.3
                        ip_address:
                            192.168.201.3
                        rtt:
                            40.0
                    2:
                        ----------
                        host_name:
                            192.168.201.3
                        ip_address:
                            192.168.201.3
                        rtt:
                            36.0
                    3:
                        ----------
                        host_name:
                            192.168.201.3
                        ip_address:
                            192.168.201.3
                        rtt:
                            40.0
    result:
        True
```

**another kind ok ping**

```
adrian@adrian-VirtualBox:~$ sudo salt 'c1_s1_cpe' net.ping 10.0.12.2
c1_s1_cpe:
    ----------
    comment:
    out:
        ----------
        success:
            ----------
            packet_loss:
                0
            probes_sent:
                5
            results:
                |_
                  ----------
```

(continues on next page)

(continued from previous page)

```
                  ip_address:
                      10.0.12.2
                  rtt:
                      0.0
              |_
                ----------
                ip_address:
                    10.0.12.2
                rtt:
                    0.0
              |_
                ----------
                ip_address:
                    10.0.12.2
                rtt:
                    0.0
              |_
                ----------
                ip_address:
                    10.0.12.2
                rtt:
                    0.0
              |_
                ----------
                ip_address:
                    10.0.12.2
                rtt:
                    0.0
          rtt_avg:
              3.0
          rtt_max:
              4.0
          rtt_min:
              1.0
          rtt_stddev:
              0.0
    result:
        True
```

**check the information about the devices**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.facts
c1_s1_cpe:
    ----------
    comment:
    out:
        ----------
        fqdn:
            r2.quadrant.edu
        hostname:
            r2
        interface_list:
            - FastEthernet0/0
            - FastEthernet0/1
            - FastEthernet1/0
        model:
            3725
```

(continues on next page)

(continued from previous page)

```
        os_version:
            3700 Software (C3725-ADVENTERPRISEK9-M), Version 12.4(15)T13, RELEASE␣
↪SOFTWARE (fc3)
        serial_number:
            FTX0945W0MY
        uptime:
            38160
        vendor:
            Cisco
    result:
        True
```

**this is interesting, you can format the output**

Salt has several out formatters, like table, json, etc

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.arp --out=table
c1_s1_cpe:
----------
    comment:
    ----------
    out:
    ----------
        ------------------------------------------------------------
        |  Age  |    Interface    |      Ip     |        Mac        |
        ------------------------------------------------------------
        | 126.0 | FastEthernet0/0 |  10.0.12.1  | C2:01:37:5C:00:01 |
        ------------------------------------------------------------
        |  0.0  | FastEthernet0/0 |  10.0.12.2  | C2:02:5D:80:00:00 |
        ------------------------------------------------------------
        |  0.0  | FastEthernet0/1 | 10.100.12.1 | C2:02:5D:80:00:01 |
        ------------------------------------------------------------
        | 149.0 | FastEthernet0/1 | 10.100.12.2 | C2:04:5F:8C:00:00 |
        ------------------------------------------------------------
```

**check arp entries**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.arp
c1_s1_cpe:
    ----------
    comment:
    out:
        |_
          ----------
          age:
              126.0
          interface:
              FastEthernet0/0
          ip:
              10.0.12.1
          mac:
              C2:01:37:5C:00:01
        |_
          ----------
          age:
              0.0
          interface:
```

(continues on next page)

(continued from previous page)

```
            FastEthernet0/0
        ip:
            10.0.12.2
        mac:
            C2:02:5D:80:00:00
      |_
        ----------
        age:
            0.0
        interface:
            FastEthernet0/1
        ip:
            10.100.12.1
        mac:
            C2:02:5D:80:00:01
      |_
        ----------
        age:
            150.0
        interface:
            FastEthernet0/1
        ip:
            10.100.12.2
        mac:
            C2:04:5F:8C:00:00
  result:
      True
```

**check interfaces**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.interfaces
c1_s1_cpe:
    ----------
    comment:
    out:
        ----------
        FastEthernet0/0:
            ----------
            description:
                to_r1
            is_enabled:
                True
            is_up:
                True
            last_flapped:
                -1.0
            mac_address:
                C2:02:5D:80:00:00
            mtu:
                1500
            speed:
                10
        FastEthernet0/1:
            ----------
            description:
                to_inside
            is_enabled:
```

(continues on next page)

(continued from previous page)

```
                True
            is_up:
                True
            last_flapped:
                -1.0
            mac_address:
                C2:02:5D:80:00:01
            mtu:
                1500
            speed:
                10
        FastEthernet1/0:
            ----------
            description:
            is_enabled:
                False
            is_up:
                False
            last_flapped:
                -1.0
            mac_address:
                C2:02:5D:80:00:10
            mtu:
                1500
            speed:
                100
    result:
        True
```

**check ip addresses of interfaces**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.ipaddrs
c1_s1_cpe:
    ----------
    comment:
    out:
        ----------
        FastEthernet0/0:
            ----------
            ipv4:
                ----------
                10.0.12.2:
                    ----------
                    prefix_length:
                        24
        FastEthernet0/1:
            ----------
            ipv4:
                ----------
                10.100.12.1:
                    ----------
                    prefix_length:
                        24
    result:
        True
```

**check arp entries**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.arp
c1_s1_cpe:
    ----------
    comment:
    out:
        |_
          ----------
          age:
              130.0
          interface:
              FastEthernet0/0
          ip:
              10.0.12.1
          mac:
              C2:01:37:5C:00:01
        |_
          ----------
          age:
              0.0
          interface:
              FastEthernet0/0
          ip:
              10.0.12.2
          mac:
              C2:02:5D:80:00:00
        |_
          ----------
          age:
              0.0
          interface:
              FastEthernet0/1
          ip:
              10.100.12.1
          mac:
              C2:02:5D:80:00:01
        |_
          ----------
          age:
              153.0
          interface:
              FastEthernet0/1
          ip:
              10.100.12.2
          mac:
              C2:04:5F:8C:00:00
    result:
        True
```

**check the same arp entries but with an "table" output formatter**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.arp --out=table
c1_s1_cpe:
----------
    comment:
    ----------
    out:
    ----------
```

(continues on next page)

(continued from previous page)

```
        ------------------------------------------------------------
        |  Age  |    Interface    |      Ip     |        Mac        |
        ------------------------------------------------------------
        | 130.0 | FastEthernet0/0 |  10.0.12.1  | C2:01:37:5C:00:01 |
        ------------------------------------------------------------
        |  0.0  | FastEthernet0/0 |  10.0.12.2  | C2:02:5D:80:00:00 |
        ------------------------------------------------------------
        |  0.0  | FastEthernet0/1 | 10.100.12.1 | C2:02:5D:80:00:01 |
        ------------------------------------------------------------
        | 154.0 | FastEthernet0/1 | 10.100.12.2 | C2:04:5F:8C:00:00 |
        ------------------------------------------------------------
    result:
    ----------
```

**or with json formatter**

```
adrian@lmint2:~$ sudo salt 'c1_s1_cpe' net.arp --out=json
{
    "c1_s1_cpe": {
        "out": [
            {
                "interface": "FastEthernet0/0",
                "mac": "C2:01:37:5C:00:01",
                "ip": "10.0.12.1",
                "age": 130.0
            },
            {
                "interface": "FastEthernet0/0",
                "mac": "C2:02:5D:80:00:00",
                "ip": "10.0.12.2",
                "age": 0.0
            },
            {
                "interface": "FastEthernet0/1",
                "mac": "C2:02:5D:80:00:01",
                "ip": "10.100.12.1",
                "age": 0.0
            },
            {
                "interface": "FastEthernet0/1",
                "mac": "C2:04:5F:8C:00:00",
                "ip": "10.100.12.2",
                "age": 154.0
            }
        ],
        "result": true,
        "comment": ""
    }
}
```

**send messages with slack**

Edit and try the playbook "backup-msg-slack.yml". After the backup you will receive a message in Slack telling you so. This is just an example of what you can do with chatOps.

Finally, check the respective project pages because this is only an intro... there are thousands of cool stuff to do.

CHAPTER 6

# Scripts

I will create some other netor-* scripts in order to make it easier to use them. Just as a mask, because at the end you should know how to work directly with them.

Ansible is very easy and straight forward, but Salt is very hard to get use to the sintaxis.

- `netor-config` to configure neto_home_directory.
- `netor-db-customers` to operate on customer table.
- `netor-db-devices` to operate on dbdevices.
- `netor-db-export` to import a CSV file to the current DB (same as `dblist -e`)
- `netor-db-import` to import a CSV file to the current DB.
- `netor-db-list` to list full BD or to export to .CSV if only the '-e' parameter is used.
- `netor-db-push` to push DB content to Ansible, Salt and bash scripts.
- `netor-db-sites` to operate on dbsites.
- `netor-db-switch` to switch DB in use in scripts.
- `netor-ping` to resolve name->ip using Ansible hosts inventory file.
- `netor-salt-restart` to restart Salt daemons.
- `netor-salt-start` to start Salt daemons.
- `netor-salt-stop` to stop Salt daemons.
- `netor-traceroute` to resolve name->ip using Ansible hosts inventory file.
- `netor-salt-view-event-bus` shortcut to execute native salt command to view event bus.

CHAPTER 7

# Modules

## 7.1 netorconf module

netorconf.**_backup_filename**(*new_netor_home_directory*, *filename*)
Create a backup of the specified configuration file

**Parameters**

- **new_netor_home_directory** – it is the actual new Neto home directory to be updated on files
- **filename** – file name to backup

**Returns** nothing

netorconf.**_create_master_config_file**(*new_netor_home_directory*, *filename*)
Create new Salt master configuration file.

**Parameters**

- **new_netor_home_directory** – it is the actual new Neto home directory to be updated on files
- **filename** – filename to backup

**Returns** nothing

netorconf.**_create_minion_config_file**(*new_netor_home_directory*, *filename*)
Create Salt minion configuration file.

**Parameters**

- **new_netor_home_directory** – Location where the file will be located
- **filename** – file name

**Returns** nothing

netorconf.**_create_proxy_config_file**(*new_netor_home_directory*, *filename*)
Create Salt proxy configuration file.

**Parameters**

- **new_netor_home_directory** – Location where the file will be located
- **filename** – file name

**Returns**

netorconf.**_create_update_master_minion_proxy**(*new_netor_home_directory*, *filename*)
Update or create (if do not exists) Salt configuration files.

**Parameters**

- **new_netor_home_directory** – it is the actual new Neto home directory to used in the process
- **filename** – file name to update

**Returns** nothing

netorconf.**_file_create_redirect**(*new_netor_home_directory*, *filename*)
Create the configuration files.

**Parameters**

- **new_netor_home_directory** – it is the actual new Neto home directory where to create the file
- **filename** – file name to create

**Returns** nothing

netorconf.**_file_update_redirect**(*new_netor_home_directory*, *filename*)
Update the configuration files.

**Parameters**

- **new_netor_home_directory** – Directory where the files are located
- **filename** – file name to update

**Returns** nothing

netorconf.**_netor_config**()
It is used for updating the Neto home directory in the configuration files and scripts.

This is useful, if you want to have 2 working installations of Neto in completely independent directories.

It will update the NETOR_HOME_DIRECTORY variable in the netor.conf file, and also in the following Neto python scripts which then works with the TinyDB: # netor/tinydb/scripts/listdb.py # netor/tinydb/scripts/pushcustdb.py # netor/tinydb/scripts/worker.py # netor/tinydb/scripts/switchdb.py

Later it will also update the hosts_file variable in the following bash scripts: # bin/netor-ping # bin/netor-traceroute

**Returns** nothing

netorconf.**_update_ansible**(*netor_home_directory*)
Update Ansible configuration files.

**Parameters** **netor_home_directory** – Neto home directory to used for updating the configuration files

**Returns** nothing

netorconf.**_update_config**(*tinydb_log_file*, *__file__*, *new_netor_home_directory*)
Execute the actual updates in the files. Salt master, minion and proxy.

**Parameters**

- **tinydb_log_file** – the filename to send the logging message after the operation is completed
- **__file__** – script name who is sending the message to log
- **new_netor_home_directory** – it is the actual new Neto home directory to be updated on files

**Returns** nothing

netorconf.**_update_master_config_file**(*new_netor_home_directory*, *filename*)
Update Salt master configuration file.

**Parameters**

- **new_netor_home_directory** – Location where the file is located
- **filename** – file name

**Returns** nothing

netorconf.**_update_minion_config_file**(*new_netor_home_directory*, *filename*)
Update Salt minion configuration file.

**Parameters**

- **new_netor_home_directory** – Location where the file is located
- **filename** – file name

**Returns**

netorconf.**_update_proxy_config_file**(*new_netor_home_directory*, *filename*)
Update Salt proxy configuration file.

**Parameters**

- **new_netor_home_directory** – Directory where the file is located
- **filename** – file name

**Returns**

netorconf.**check_netor_config**(*netor_home_directory*)
Verifies if the `netor.config` file exists in the file tree.

**Parameters** **netor_home_directory** – to verify if the netor home directory and file exists

**Returns** nothing

netorconf.**replace_static_vars_scripts**(*filename*, *search*, *replace*, *delimiter*, *extra*)
Replace line by line the `NETOR_HOME_DIRECTORY` static variable in scripts.

**Parameters**

- **filename** – filename to review
- **search** – search pattern to look for
- **replace** – patter to replace
- **delimiter** – to add a delimiter surrounding the path names
- **extra** – add extra path information

**Returns** nothing

## 7.2 dbparam module

**class** dbparam.**DbParam**(*db_path_name*)
Bases: object

Startup common DB parameters and tables

## 7.3 worker module

worker.**_redirect**()
Worker redirect to all operations on the DB tables.

Operates the DB on table specified, for list, add, modify and delete registers.

It uses the local static variables of the worker.py script as NETOR_HOME_DIRECTORY and DB_PATH_NAME, unless a full path name to a TinyDB database is specified.

It supports the following parameters:

1. specifying operations:
   - l (list)
   - a (add)
   - m (modify)
   - d (delete)
2. table to operate with
   - customer
   - sites
   - devices
3. specifying DB full path name

Example: ../tinydb/scripts/worker.py -l customers /full/path/name/database.json

Logging to file ./log/tinydb.log

**Returns** nothing

## 7.4 customers module

**class** customers.**Customers**(*db_path_name*)
Bases: *dbparam.DbParam*

Customer worker to operate DB on table customers (list, add, modify and delete methods).

Customer name has to be alphanumeric and less than 20 characters. At the DB all the spaces " " will be replaced by underscores "_".

**static _check_value**(*value*)
Verifies the value to be alphanumeric string, and less than 20 characters :param value: string to verify :return: True if the value is a valid string otherwise returns False

**add**()
Add a *customer*.

**Returns** `Customer` name added or `False` if the customer already exists.

**delete**()
Delete a *customer*.

**Returns** `Customer` name deleted or `False` if the customer already exists.

**list**()
List the table `customers` ordered alphabetically.

**Returns** True

**modify**()
Modify a *customer*.

**Returns** `Customer` original name and `new_name`, or `False` if the customer do not exists.

## 7.5 sites module

**class** sites.**Sites**(*db_path_name*)
Bases: *dbparam.DbParam*

Sites worker to operate DB on table sites (list, add, modify and delete methods).

Site name has to be alphanumeric and less than 20 characters. At the DB all the spaces `" "` will be replaced by underscores `"_"`.

**static _check_value**(*value*)
Verifies the value to be alphanumeric string, and less than 20 characters.

**Parameters** **value** – string to verify

**Returns** `True` if the value is a valid string otherwise returns `False`

**add**()
Add a *site*.

**Returns** `Site` name added or `False` if the parent Customer do not exists, or the site already exists.

**delete**()
Delete a *site*.

**Returns** `Site` name deleted or `False` if the site do not exists.

**list**()
List the table `sites` ordered alphabetically, and filtered by customer if that option was selected.

**Returns** `True` if the site was listed or `False` if the customer do not exists.

**modify**()
Modify a *site*.

**Returns** `Site` original name and `new_name` name, or `False` if there is with parent Customer or site

## 7.6 devices module

**class** devices.**Devices**(*db_path_name*)
Bases: *dbparam.DbParam*

Devices worker to operate DB on table `devices` (list, add, modify and delete methods).

Device names has to be alphanumeric and less than 20 characters. At the DB all the spaces `" "` will be replaced by underscores `"_"`.

**_OS_LIST = ['ios', 'iosxr', 'nxos', 'eos', 'junos', 'f5', 'win', 'linux']**

**static _check_value**(*value*)
Verifies the value to be alphanumeric string and less than 20 characters.

**Parameters** **value** – string to verify

**Returns** `True` if the value is a valid string otherwise returns `False`

**add**()
Add a *device*.

**Returns** `Dev_name` added or `False` if there was an error.

**delete**()
Delete a *device*.

**Returns** `dev_name` of the delete device, or `False` it there was an error.

**list**()
List the table `devices` ordered alphabetically, and filtered according the selected options (customer, site, IP, device name, or OS.

**Returns** `True` if the devices were listed, or `False` if there is no match between the conditions selected.

**modify**()
Modify a *device*.

**Returns** `Dev_name` original and `new_dev_name` of the modified device, or `False` if there was an error.

## 7.7 listdb module

**class** listdb.**DB**(*db_path_name*)
Bases: *dbparam.DbParam*

**export_csv**(*tinydb_log_file*, *db_path_name*, *export_path_name*)
Export full DB table 'devices' content to CSV

**Parameters**

- **tinydb_log_file** – log file to send the message
- **db_path_name** – full path name of the DB to list
- **export_path_name** – full path name of .CVS file to export the DB

**Returns**

**list**(*tinydb_log_file*, *db_path_name*)
List full DB content.

**Parameters**

- **tinydb_log_file** – log file to send the message
- **db_path_name** – full path name of the DB to list

**Returns**

listdb.**_listdb**()

List the full content of the DB specified in the configuration file `netor.conf`, unless a full path name to different TinyDB database is specified as an argument.

Verify existence of `netor.conf` file, since it could be required by the confparse module in order to get the DB full path name.

**example:** `../tinydb/scripts/listdb.py /full/path/name/database.json`

Also has an export function to dump the current DB to a `.csv` file. If you want to export another DB you will need to `netor-db-switch` first.

**example:** `../tinydb/scripts/listdb.py -e`

**Returns** nothing

## 7.8 switchdb module

switchdb.**_switchdb**()

Updates the full path name of the DB in configuration and scripts files.

As an example it can be used to have multiple database files on the same netor home directory, in order to support different group of devices, like different customers, environments, group of devices, or even for copying DB files to different folders or machines for sharing them.

**Returns** nothing

switchdb.**_update_config**(*full_db_path_name*)

Updates the configuration files.

*To complete the process you need to push the DB with netor-db-push.*

**Parameters** **full_db_path_name** – full path name of the DB to use

**Returns** nothing

## 7.9 pushcustdb module

**class** pushcustdb.**DB**(*db_path_name*)

Bases: `object`

Class DB with parameters used by all the methods to push DB info into **Ansible** and **Salt**.

**static ansible_push_inventory**(*tinydb_log_file*, *ansible_hosts_path_name*, *ansible_backup_hosts_path_name*, *devices_to_push*)

Backup of the `ansible/hosts` inventory file to `ansible/backup` directory.

Generates a new `ansible/hosts` file based on the DB information (actual push of data).

**Parameters**

- **tinydb_log_file** – log file to send the message after the operations are being completed
- **ansible_hosts_path_name** – full path name of the ansible hosts file
- **ansible_backup_hosts_path_name** – full path name of the ansible hosts file directory
- **devices_to_push** – list of devices to push

**Returns** nothing

**static salt_push_inventory**(*tinydb_log_file*, *salt_minion_path_name*, *salt_backup_directory*, *salt_pillar_directory*, *salt_backup_pillar_directory*, *salt_top_path_name*, *salt_states_directory*, *devices_to_push*)

Backup `salt/config/minion` file to `salt/config/backup` directory.

Generates a new minion file based on the actual minion file, but replacing the first lines **(to be improved in order to review the content/update of the lines)**.

Generates the new `top.sls` file with the devices listed in the DB, and all the states only in the '*' section.

**(To be improved to actually add the corresponding states to specific devices)**.

Backup `salt/config/pillar/\*.sls` files to `/salt/config/backup/pillar directory`.

Generates new `pillar/*device_name*.sls` files based on the content of the DB file, if the device has to be managed via `salt-proxy` according the DB `salt_proxy_required` field. With the following details:

```
proxy:
  proxytype: napalm
  driver: ios
  host: dev_ip
  username: userid
  password: passwd
  optional_args:
    use_keys: True
    auto_rollback_on_error: True
```

**Parameters**

- **tinydb_log_file** – log file to send the message after the operations are being completed
- **salt_minion_path_name** – minion full path name
- **salt_backup_directory** – minion backup directory
- **salt_pillar_directory** – pillar .sls files directory
- **salt_backup_pillar_directory** – backup pillar *.sls directory
- **salt_top_path_name** – full path name of ''top.sls" file
- **salt_states_directory** – add all the states to the general `\'*'` section of the `top.sls` file
- **devices_to_push** – list of devices to push to Ansible and Salt

**Returns** nothing

**select_devices_to_push**(*filter_expression*)
Provides a mechanism to filter the devices in the DB before pushing and replacing Ansible and Salt configuration.

**Parameters** **filter_expression** – expression to use a the filter. Support RegEx.

**Returns** db_devices with the filtered device list

pushcustdb.**_start_process**()
Checks if the netor.conf file exist according the local static variable _NETOR_HOME_DIRECTORY.

It supports passing as parameter the _NETOR_HOME_DIRECTORY in order to push to the DB specified in the netor.conf configuration file.

Then execute the functions to push the DB content on **Ansible** ans **Salt** configuration files, according to the settings of the netor.conf configuration file.

It supports filtering of the DB, with literal text or RegEx, in order to not import more devices than required. As an example, you could have several clients on a DB, and push only the clients with which you are working, or the same example applies to work per sites, per IP range, per device name, etc.

**Restart of the Salt minion is required.**

**Returns** nothing

## 7.10 importcsv module

**class** importcsv.**ImportDevices**(*db_path_name*)
Bases: *dbparam.DbParam*

Class to import data from a CSV file

Format: customer,site,dev_name,dev_ip,os,userid,passwd,salt_proxy_required

Names should be alphanumeric, and less than 20 characters, all names will be transformed to lowercase.

**add_line_to_db**(*checked_line_values*, *line_number*)
Add a customers, sites and devices to the DB.

**Parameters**

- **checked_line_values** – CSV line to import to DB.
- **line_number** – line number being processed.

**Returns** True if the lines was added and False if not.

importcsv.**_check_values**(*line_value_list*)
Verify the values of each field of the line being processed.

**Parameters** **line_value_list** – contains the line transformed to a list

**Returns** line_value_list with corrections, or False if the list has an invalid value.

importcsv.**_import_csv**(*destination_db_to_import*, *csv_file_to_import*)
Import the CSV to the DB

**Parameters**

- **destination_db_to_import** – DB to import the CSV
- **csv_file_to_import** – the file to import to the DB

**Returns** total_lines_number_processed and lines_imported

importcsv.**start_process**()
Starts the process of import the CSV to DB, asking for the path names to the files.

**Returns** Nothing

## 7.11 tinydblogging module

CHAPTER 8

# TODOs

- ADD ENCRYPTION TO STORE THE USER ID PASSWORD IN TINYDB
- Create netor-update script to update the software
- Redo netorconf.py.
- Auto testing.
- Upload to PyPi and work in adapting de structure.
- Reformat code to make it reusable and with less repeated code.
- Work on bash or python scripts to mirror common Ansible and Salt operations in order to make it easier to use and start learning about them Ansible and Salt.

CHAPTER 9

# Limitations

- Tested on Linux and macOS. Don't support Windows, since Ansible and Salt do not support them.
- Only supports Python 3.
- If you change you `netor_home_directory` you have to update the PATH environment variable in order to look for the scripts in the correct folder.

CHAPTER 10

# Thank you notes

These passionate individuals that are always there to help, teach and guide us.

- Python3 for network engineers: with Kirk's online trainings I started this journey of learning Python, with network orientation: Kirk Byers <ktbyers@twb-tech.com>
- NAPALM: David Barroso dbarrosop@dravetech.com, Mircea Ulinic ping@mirceaulinic.net, and Kirk Byers ktbyers@twb-tech.com
- TinyDB: Markus Siemens markus@m-siemens.de
- The authors of these great books which helped me a lot:
  - Network Programmability and Automation (Jason Edelman, Scott S. Lowe and Matt Oswalt)
  - Network Automation at Scale (Mircea Ulinic and Seth House)
- Ansible and Salt teams
- OpenSource community in general

CHAPTER 11

# Indices and tables

- genindex
- modindex
- search

# Python Module Index

# Index

## Symbols



## A



## C



## D



## E



## I



## L



## M

## N



## P



## R



## S



## W