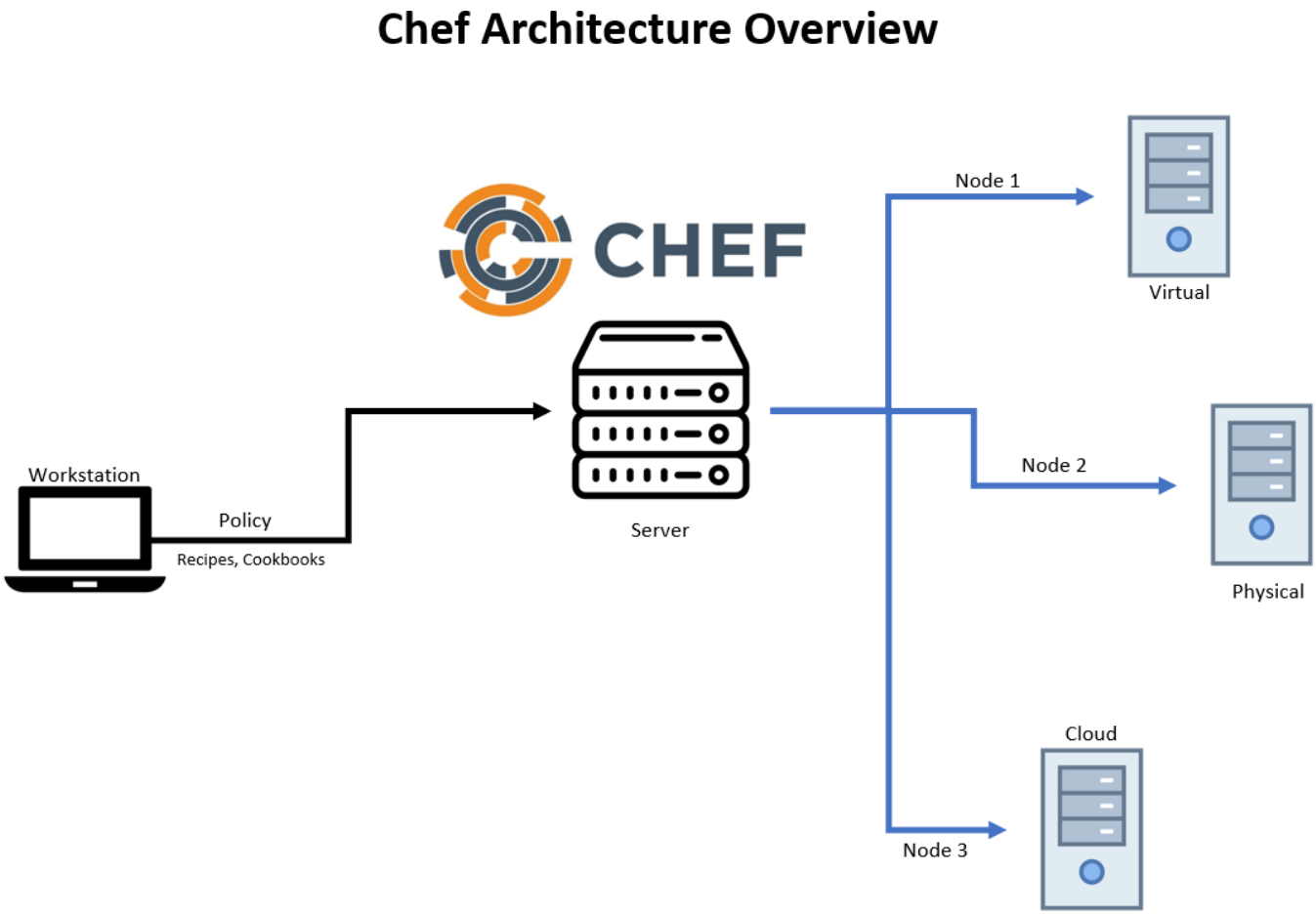
Installing Chef Server Workstation on Ubuntu 18.04

Chef is a Ruby based configuration management tool used to define infrastructure as code. This enables users to automate the management of many *nodes* and maintain consistency across those nodes. *Recipes* declare the desired state for managed nodes and are created on a user's *workstation* using the *Chef Workstation* package. Your recipes are distributed across nodes via a *Chef server*. **A *Chef client*, installed on each node**, is in charge of applying the recipe to its corresponding node.

This guide will show you how to create and configure a Chef server and workstation on Ubuntu 18.04. You'll also learn how to bootstrap a node to manage with Chef. This will involve setting up three machines (Chef Server, Chef Workstation, and Node).



Prerequisites

1. **Chef Server**

The **Chef Server** is the central hub where all your infrastructure configurations are stored. It holds the recipes and cookbooks that define how machines should be configured, and it communicates with Chef Clients to apply these configurations to nodes.

2. **Chef Workstation**

The **Chef Workstation** is where you create and manage the code (recipes and cookbooks) that will be applied to nodes. It's the machine where you write, test, and upload configurations to the Chef Server for deployment.

3. **Chef-client**

The **Chef-client** runs on each node (machine) that Chef manages. It pulls configuration data from the Chef Server and ensures that the node is in the desired state, applying any necessary changes to match the defined recipes.

Ensure Workstation and Server are running **Ubuntu 18.04** and that they are up-to-date:

```
sudo apt update && sudo apt upgrade
```

Chef Server Installation

Install Chef Server

1. Download the latest Chef Server core package for Ubuntu 18.04:

```
wget https://packages.chef.io/files/stable/chef-server/13.0.17/ubuntu/18.04/chef-server-core_13.0.17-1_amd64.deb
```

2. Install Chef Server:

```
sudo dpkg -i chef-server-core_*.deb
```

3. Clean up the downloaded package: (Optional)

```
rm chef-server-core_*.deb
```

4. Reconfigure Chef Server to start the services:

```
sudo chef-server-ctl reconfigure
```

Create Chef User and Organization

Create a user and an organization that will be associated with the Chef Server.

1. Create a `.chef` directory to store keys:

```
mkdir ~/.chef
```

2. Create an administrator user for Chef:

```
sudo chef-server-ctl user-create USER_NAME FIRST_NAME LAST_NAME EMAIL 'PASSWORD' -
-filename ~/.chef/USER_NAME.pem
```

Eg >

```
chef-server-ctl user-create harry "Harry" "Potter" harry.potter@hogwarts.com
"Harry@123" -f /root/admin.pem
```

3. Create an organization and associate the user with it:

`ORG_NAME` must be in all lower case.

```
sudo chef-server-ctl org-create ORG_NAME "Full Organization Name" --
association_user USER_NAME --filename ~/.chef/ORG_NAME.pem
```

> Replace `USER_NAME`, `ORG_NAME`, `FIRST_NAME`, `LAST_NAME`, `EMAIL`, and `PASSWORD` with your own details.

Eg>

```
chef-server-ctl org-create hogswarts "Hogwarts" --association_user harry -f
/root/myorg-validator.pem
```

When you create user and organization, it will generate two .pem files under .chef folder, which is present in the home directory, to view the folder type command (ls -a) under the home directory.

When using the `user-list` or `org-list` commands, errors can occur if the Chef server is not properly configured or there are issues with connectivity, permissions, or server configuration. Here's how you can handle potential errors:

1. Error Handling for `user-list` and `org-list` Commands

- **Error Logs**: When running `user-list` or `org-list`, if you encounter an error, you should check the Chef server logs for more details. The logs can be found in the following location:
  - `/var/log/chef-server/` (Chef server logs).
  - `/var/log/opscode/` (This directory holds the Chef server log files for different components).

  Use commands like `tail -f` or `cat` to view logs, such as:

  ```
  tail -f /var/log/chef-server/chef-server.log
  ```

- **Permissions**: Ensure you have the correct permissions to access the Chef server. Sometimes, a failure occurs because of insufficient privileges or a lack of proper authentication.
- **Authentication Issues**: If you have problems with authentication or access to the Chef server, make sure your `.pem` files are valid and properly configured. Verify that the `USER_NAME.pem` file exists in the `~/.chef/` directory, which is needed for proper authentication.

2. .chef Folder and the .pem Files

- **Location of .chef Folder**: The `.chef` folder is typically located in the home directory of the user running the Chef commands. You can check its existence by running:

```
ls -a ~/
```

- **.pem Files**: When you create a Chef user or an organization, the system generates `.pem` files for authentication purposes. These files are typically stored in the `.chef` directory. If you see files like `USER_NAME.pem` and `ORG_NAME.pem`, it means the user and organization were successfully created.

Verify Users and Organizations

You can list the users and organizations on your Chef server using:

```
sudo chef-server-ctl user-list
sudo chef-server-ctl org-list
```

You should see in output you created user and pivotal user

```
harry
pivotal
```

and in org list you should see your created organization

```
hogwarts
```

Chef Workstation Setup

Install Chef Workstation

1. Download the Chef Workstation package:

```
wget https://packages.chef.io/files/stable/chef-
workstation/0.2.43/ubuntu/18.04/chef-workstation_0.2.43-1_amd64.deb
```

2. Install Chef Workstation:

```
sudo dpkg -i chef-workstation_*.deb
```

3. Clean up the downloaded package:

```
rm chef-workstation_*.deb
```

Create Chef Repository

1. Generate a Chef repository:

```
chef generate repo chef-repo
```

2. Move into the repository directory:

```
cd ~/chef-repo
```

3. When you create user and organization ON SERVER, it will generate two .pem files under `.chef` folder, which is present in the home directory, to view the folder type command (`ls -a`) under the home directory.
4. Send these two files to the chef-workstation in folder (`~/chef-repo/.chef/`)
5. How to send file: `scp /source/path/ username@ip:~/chef-repo/.chef/` .

Configure Knife

Knife is a command-line tool used to manage your Chef infrastructure. To configure Knife:

1. Create the `.chef` directory:

```
mkdir ~/chef-repo/.chef
```

2. Create the Knife configuration file (`knife.rb`) in the `.chef` directory:

```
nano ~/chef-repo/.chef/knife.rb
```

Add the following configuration:

```
chef_server_url 'https://SERVER_IP/organizations/ORG_NAME'
node_name 'USER_NAME' #use username that was created ON SERVER
client_key '/root/chef-repo/.chef/admin.pem' #created on Server
ssl_verify_mode :verify_none
cache_type               'BasicFile'
cache_options( :path => "#{ENV['HOME']}/.chef/checksums" )
cookbook_path            ["#{current_dir}/../cookbooks"]
```

Test Knife Configuration

Test if the configuration is working by listing the Chef clients:

```
knife client list
```

In Output you must see your org validator `hogwarts-validator` this means server and workstation are connected.

CREATE A COOKBOOK

1. Go to Chef-Workstation, go to chef-repo/cookbooks/ run command

```
chef generate cookbook cookbook_name
```

This creates a cookbook template in `chef-repo` directory with a `cookbooks/cookbook_name/recipes/default.rb` recipe in it

It is written in ruby language

By default, Chef generates a `default.rb` recipe, but you can have as many recipes as needed within the same cookbook. The name `default.rb` is a convention, but it's not mandatory. You can create additional recipes for specific tasks, and you can name them whatever you want (e.g., `install.rb`, `configure.rb`, etc.).

Is `default.rb` the only recipe we can create?

No, you are not restricted to just a `default.rb` recipe. The `default.rb` file is just a convention that Chef looks for when no specific recipe is specified. You can create multiple recipes and reference them as needed, for example:

- `install.rb` for installation-related tasks
- `configure.rb` for configuration tasks
- `patch_installation.rb` for patching tasks
-

2. **Modify the `default.rb` content:**

The `default.rb` recipe is written in Ruby language, and you can edit it using editors like `vim` or `nano`.

To upload the cookbook to the Chef server after making changes, use the following command:

```
knife cookbook upload cookbook_name
```

This will upload the cookbook to the Chef server, making it available for clients to pull.

If the command fails:

1. **Check for authentication issues**: Ensure your `knife.rb` configuration file is correct and you have the necessary permissions. If there are authentication errors, you might need to reauthenticate or check your Chef server credentials.
2. **Check network connectivity**: Ensure your workstation can communicate with the Chef server. Network issues could cause a failure to upload the cookbook.
3. **Check for missing dependencies**: Ensure all dependencies for the cookbook are available and properly configured. In your `knife.rb` file, make sure the `cookbook_path` is correctly set to point to the location of your cookbooks. The following line should be included in your `knife.rb`:

   ```
   cookbook_path ["#{current_dir}/../cookbooks"]
   ```

   This ensures that Chef is aware of the correct path to your cookbooks and can find and upload them without issues.
4. **Check cookbook name**: Ensure you enter correct cookbook name
   - If the upload fails, you will see an error message with details on what went wrong. Look for clues like:
     - "Permission denied" (authentication issues)
     - "Could not connect" (network or server issues)
     - "Cookbook validation failed" (cookbook-related issues)

If you encounter a failure, review the error message to diagnose and resolve the issue.

Bootstrap a Node (Works only if you have SSH)

Bootstrapping a node installs the Chef client on the node and validates the node. This allows the node to read from the Chef server and pull down and apply any needed configuration updates detected by the chef-client.

> If you encounter any `401 Unauthorized` errors ensure that your `ORGANIZATION.pem` file has `700` permissions. See Chef's troubleshooting guide for further information on diagnosing authentication errors.

1. From your *workstation*, navigate to your `~/chef-repo/.chef` directory:

   ```
   cd ~/chef-repo/.chef
   ```

2. Bootstrap the client node either using the client node's root user, or a user with elevated privileges:

   - **As the node's root user**, change `password` to your root password and `nodename` to the desired name for your client node. You can leave this off if you would like the name to default to your node's hostname:

     ```
     knife bootstrap 192.0.2.0 -x root -P password --node-name nodename
     ```

   - **As a user with sudo privileges**, change `username` to a node user, `password` to the user's password and `nodename` to the desired name for the client node. You can leave this off if you would like the name to default to your node's hostname:

     ```
      knife bootstrap 192.0.2.0 -x username -P password --use-sudo-password
     --node-name nodename
     ```

   - **As a user with key-pair authentication**, change `username` to a node user, and `nodename` to the desired name for the client node. You can leave this off if you would like the name to default to your client node's hostname:

     ```
     knife bootstrap 192.0.2.0 --ssh-user username --sudo --identity-file
     ~/.ssh/id_rsa.pub --node-name hostname
     ```

3. Confirm that the node has been bootstrapped by listing the client nodes:

   ```
    knife client list
   ```

   Your new client node should be included in the list.

   ```
   workstation@workstation:~/chef-repo$ knife client list
   Test-0001
   Test-0002
   hogswarts-validator
   ```

Add the cookbook to the node's run-list:

1. Go to Chef-Workstation run command

```
knife node run_list add NODE_NAME recipe[COOKBOOK_NAME]
```

(ex. knife node run_list add Test-001 recipe[example_cookbook]).

2. We have also created a script to add cookbook to run-list for number of nodes based on pattern

Use the bash script provided to you move it to workstation `chef-repo` folder

and then use this command

```
./upload_cookbook <NODE_PATTERN> <COOKBOOK_NAME>
```

eg . > ./upload_cookbook Test-00 example_cookbook

Script to do this

```
#!/bin/bash

if [ "$#" -ne 2 ]; then
        echo "command like ./upload_cookbook <NODE_PATTERN> <COOKBOOK_NAME>"
        exit
fi

# Variables
COOKBOOK_NAME=$2   # Replace with your cookbook name
PATTERN=$1     # Replace with your desired pattern (e.g., "test-")
LOG_FILE="/var/log/chef_node_update.log"

# Upload cookbook to server
knife cookbook upload $COOKBOOK_NAME

# Fetch all nodes from the Chef server
echo "Fetching node list from Chef server..."
NODE_LIST=$(knife node list)

# Loop through each node
for NODE in $NODE_LIST; do
    if [[ "$NODE" == *"$PATTERN"* ]]; then
        echo "Adding cookbook '$COOKBOOK_NAME' to run list of node: $NODE"

        # Add the cookbook to the node's run list
        knife node run_list add "$NODE" "recipe[$COOKBOOK_NAME]" >> "$LOG_FILE"
2>&1

        if [ $? -eq 0 ]; then
            echo "Successfully added '$COOKBOOK_NAME' to $NODE"
        else
            echo "Failed to add cookbook to $NODE. Check $LOG_FILE for details."
        fi
    fi
done

echo "Script execution completed."
```

The command `sudo chef-client` fetches the required package, which is added to the run-list.

To view the status of the last pull, run the command `knife status` on the workstation. This will list all nodes and their last pull times.

Output of `knife-status` should look like this

```
3 minutes ago, Test-001, ubuntu 22.04
3 minutes ago, Test-002, ubuntu 22.04
3 minutes ago, Test-003, ubuntu 22.04
```

To Setup Chef-Client in Goldmaster VM if SSH is disabled

1. This settings are for you to create a fresh client setup without using Bootstrap command because it uses SSH so now client and server will talk with HTTP (443)

CHEF CLIENT SETUP:

1. Install Chef client from the chef website.
2. This is a link to download Chef-Client (https://downloads.chef.io/chef) OR https://community.chef.io/downloads/tools/infra-client?os=ubuntu

select prefered OS and version

> You can install Ubuntu 20.04 version chef-client in 22.04 We have tested with this and it is working fine

3. Install it using command `dpkg -i *c`
4. After that go to `/etc` folder `cd /etc`, make directory `mkdir chef`, under chef folder make a file `client.rb`.

Add the following configuration:

```
chef_server_url "[https://CHEF-SERVER-IP/organizations/ORGNAME"]
validation_client_name "ORGNAME-validator"
log_location "/var/log/chef-exec.log"
ssl_verify_mode :verify_none
validation_key '/etc/chef/myorg-validator.pem'
interval 120 # pulls changes every 2 minutes
splay 20 # random delay to help server not overload
```

All activity is logged to `/var/log/chef-exec.log`. You can view the log file to check the output of `chef-client` execution:

```
tail -f /var/log/chef-exec.log
```

5. When you create user and organization ON SERVER, it will generate two .pem files under `.chef` folder, which is present in the home directory, to view the folder type command (`ls -a`) under the home directory.
6. Send these two files to the chef client in folder (`~/etc/chef/`)
7. How to send file: `scp /source/path/ username@ip:~/etc/chef/`.

Create a cronjob to connect node automatically on reboot

To add a cronjob to `/etc/crontab` for running `chef-client` on system startup follow these steps:

1. Open `/etc/crontab` using a text editor:

```
sudo vim /etc/crontab
```

2. Add the following lines:

   At first boot (delays `chef-client` by 180 seconds to allow the system to stabilize):

```
@reboot (sleep 180; chef-client)
```

3. Save the file and exit the editor.

- On the **first boot**, the cronjob triggers the `chef-client` with a 180-second delay to allow the system to stabilize. This process will authenticate the node, create `client.pem`, and connect to the Chef server for the initial setup.

Create a cronjob to pull changes if you have not used interval and splay in client.rb

1. Open `/etc/crontab` using a text editor:

```
sudo nano /etc/crontab
```

2. **Every 30 minutes** (executes `chef-client` to pull configuration changes from the Chef server):

```
`*/30 * * * * chef-client
```

3. Save the file and exit the editor.

What happens:

- Every **30 minutes** thereafter, the cronjob triggers `chef-client` to check for configuration changes and apply them.

This setup ensures that chef-client runs on startup (with a delay) and periodically every 30 minutes, keeping the system synchronized with the Chef server.