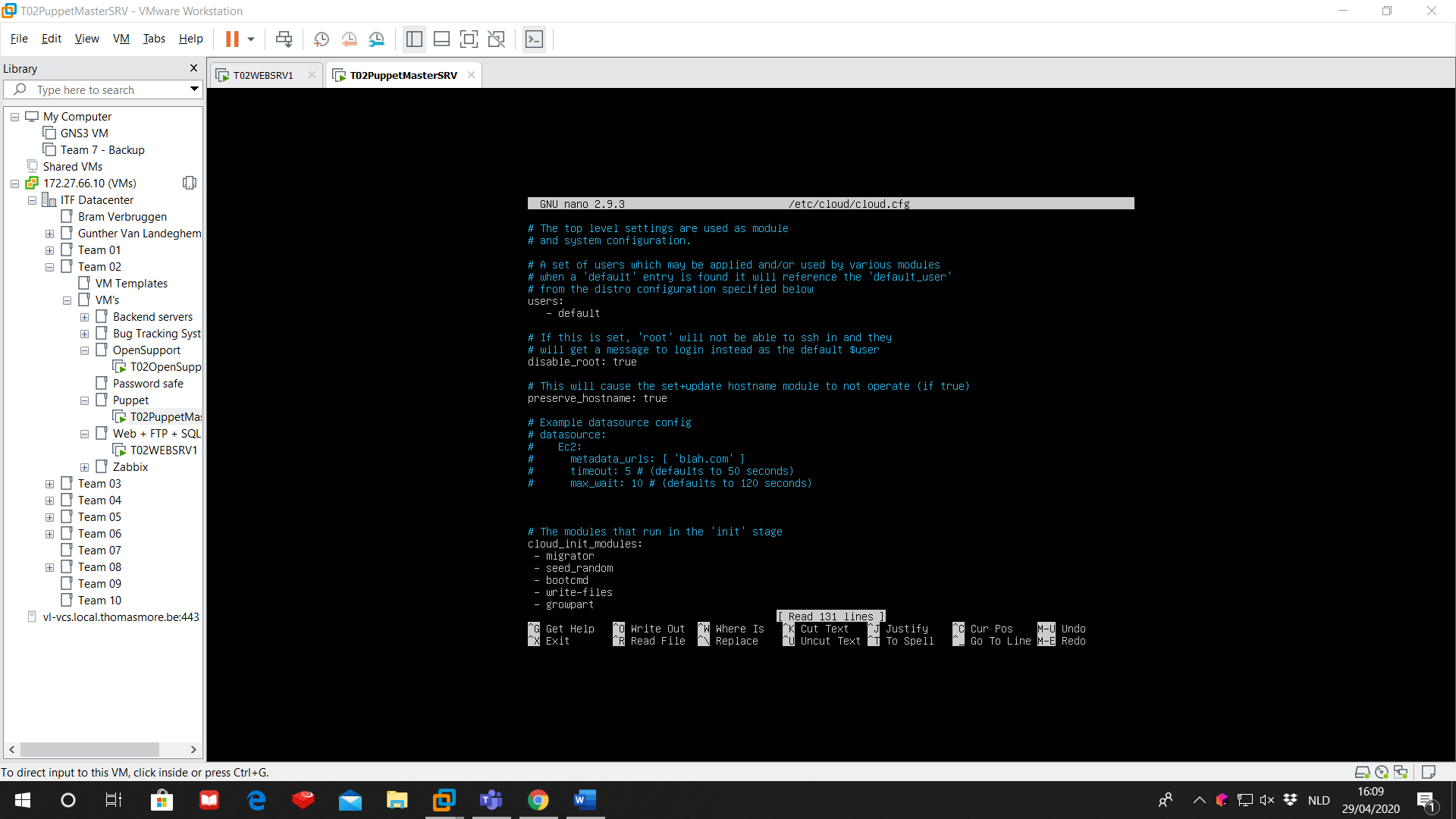
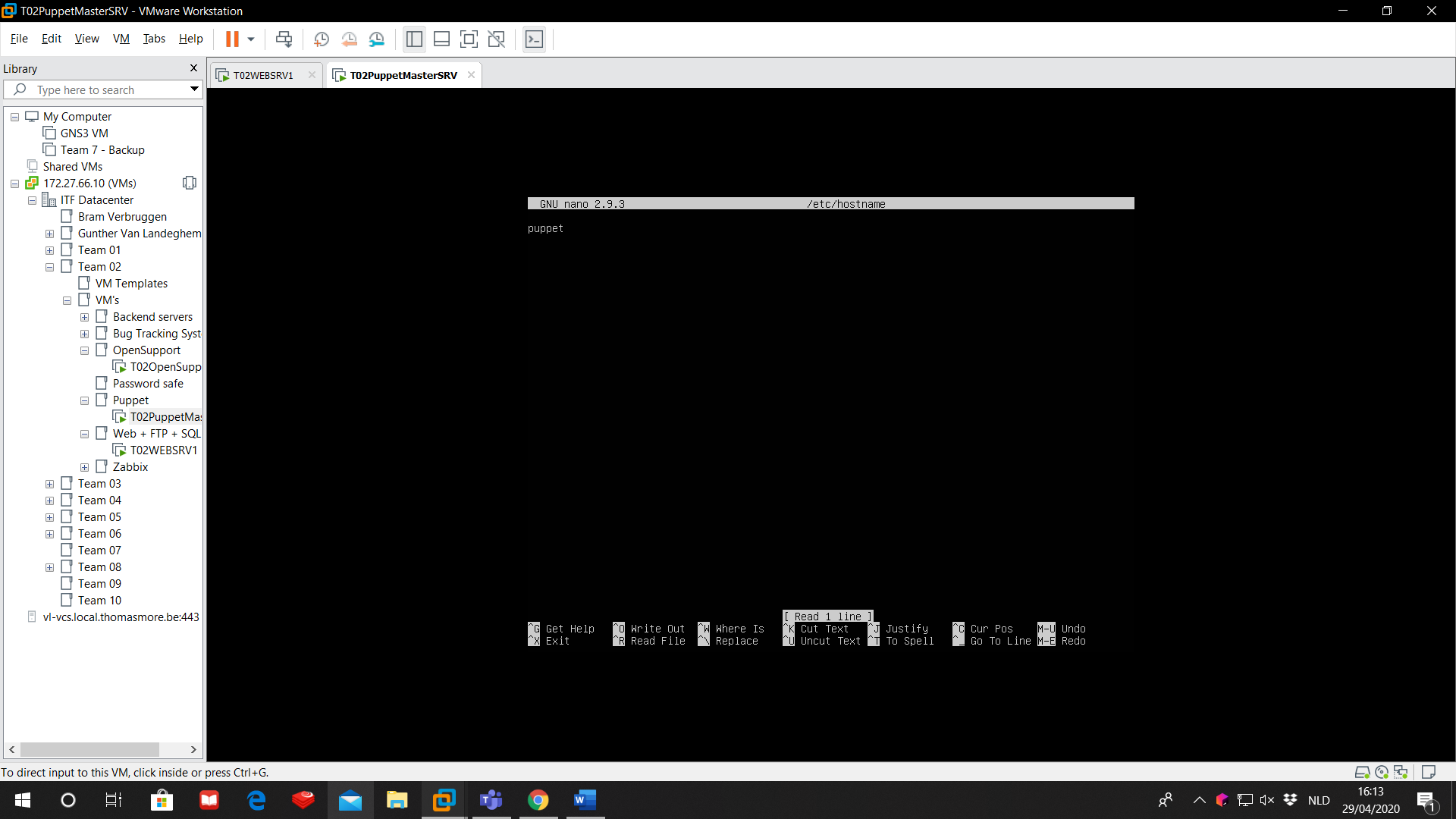
# Puppet user manual

1. Prerequisites
   1. Hostnames
      1. Master

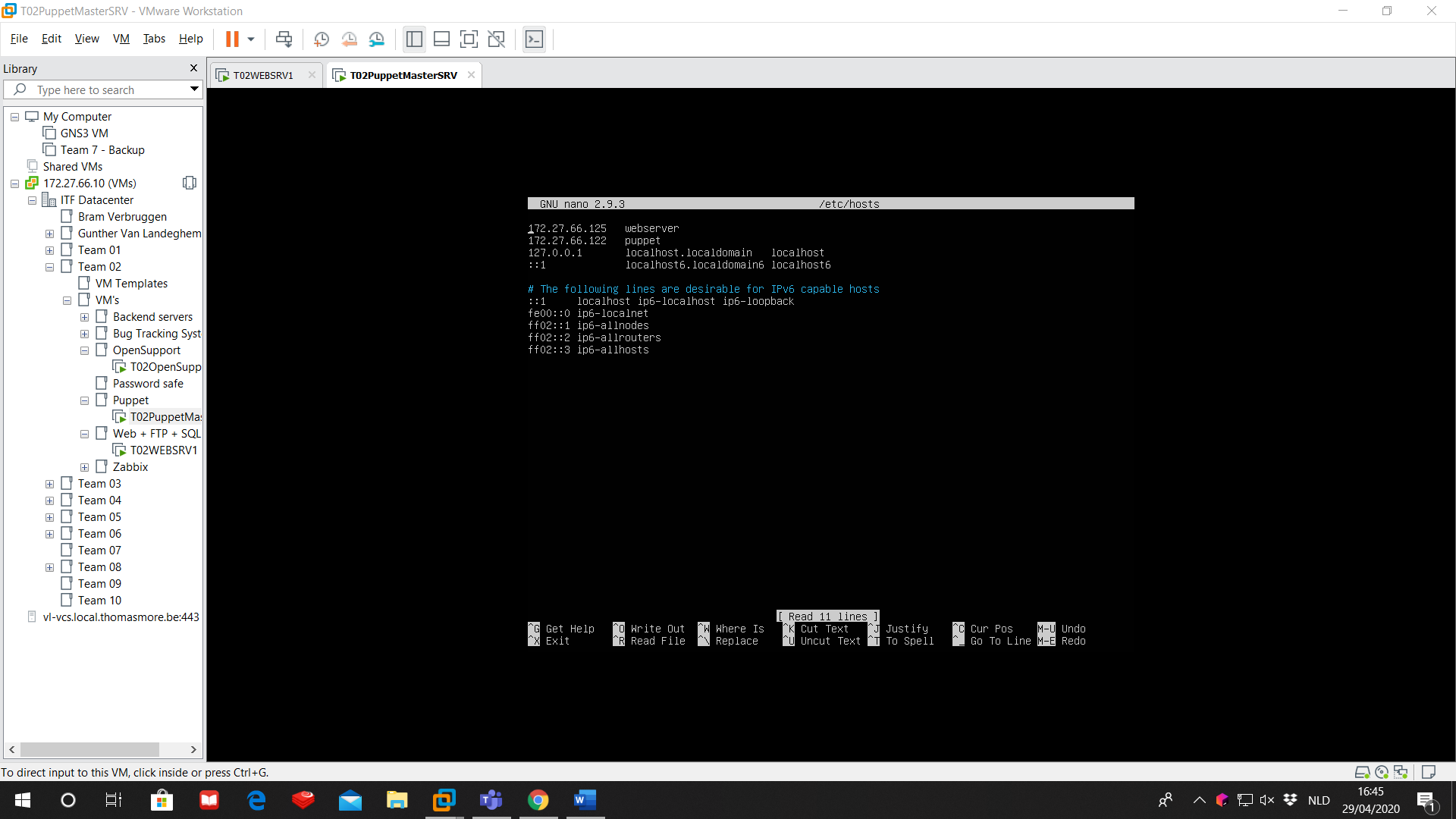
The puppet agents locate their master on it’s host name. By default they’ll search for the hostname ‘puppet’, thus we need to change the hostname of the puppet master.

First we change /etc/coud/cloud.cfg to preserve the new hostname.



Next, we change the hostname in /etc/hostname to ‘puppet’

Finally we make a new entry in /etc/hosts with the IP of the puppet master (=the IP of this machine) and it’s hostname (‘puppet’). We will also add an entry for each agent we will use. For this example we will add our webserver with IP 172.27.66.125 and hostname ‘webserver’.

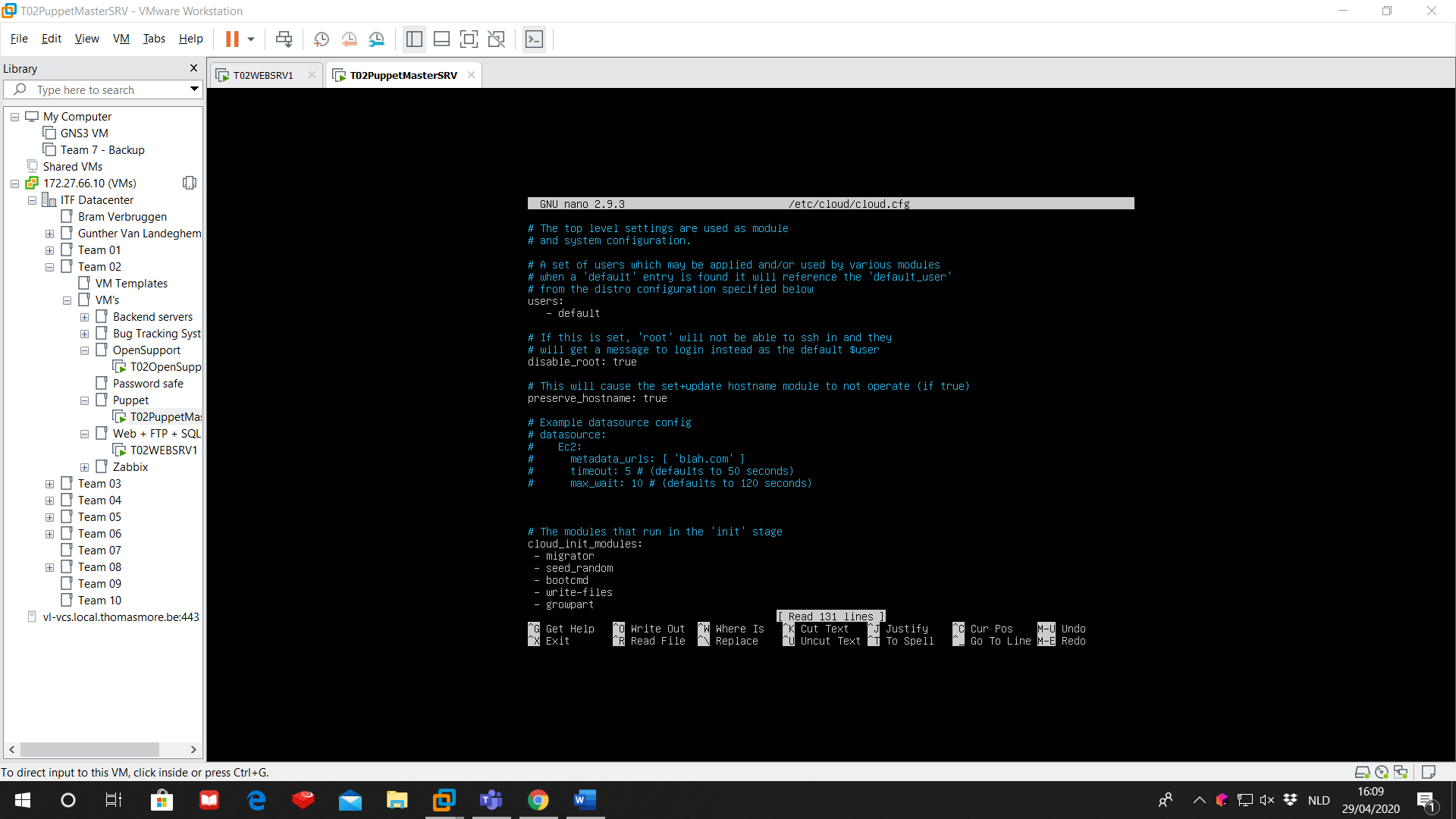


Now reboot the machine to save the new host settings.

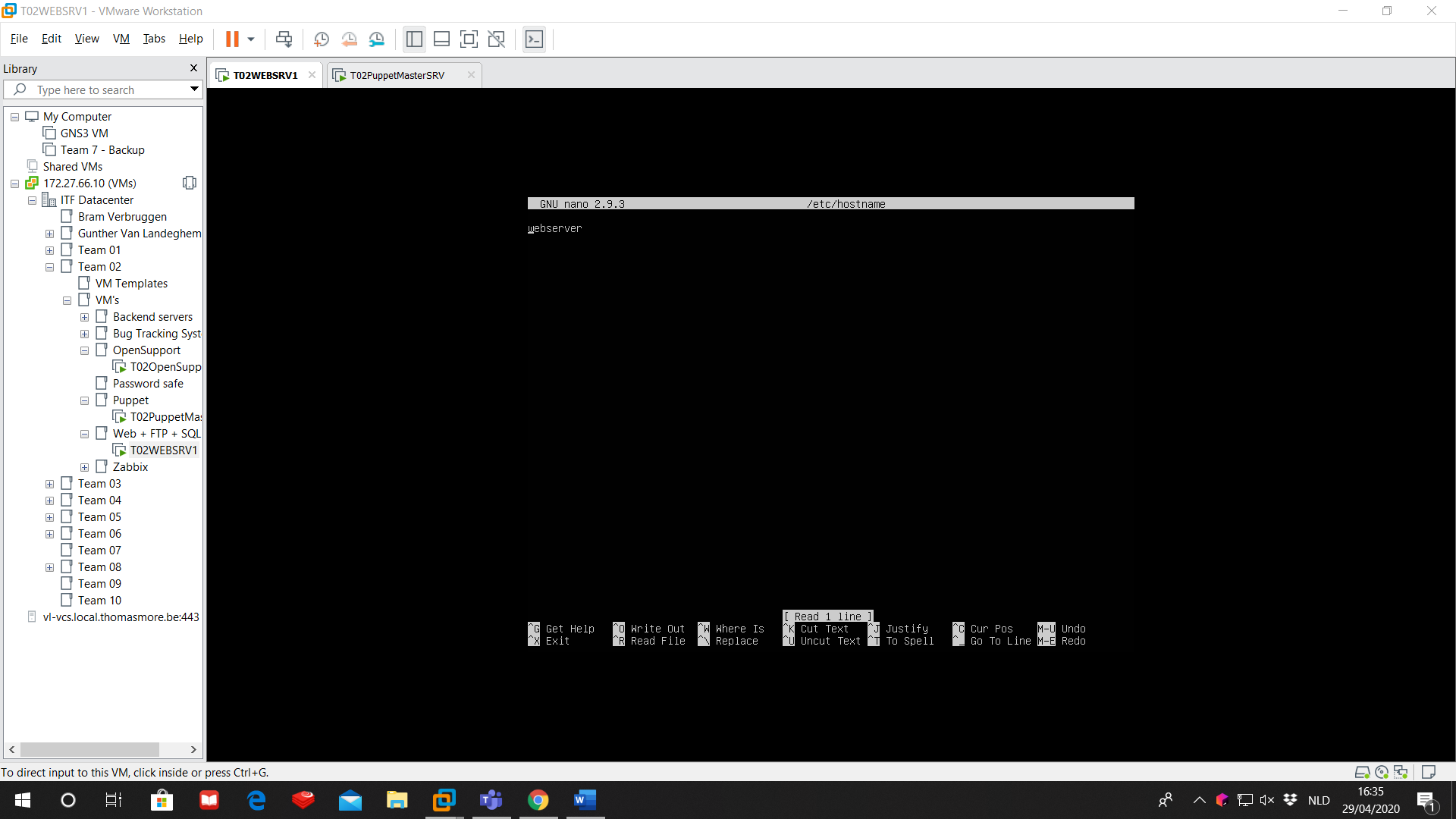
* + 1. Agents

For every agent we will install we will have to do the same thing. We will use the webserver agent as example.

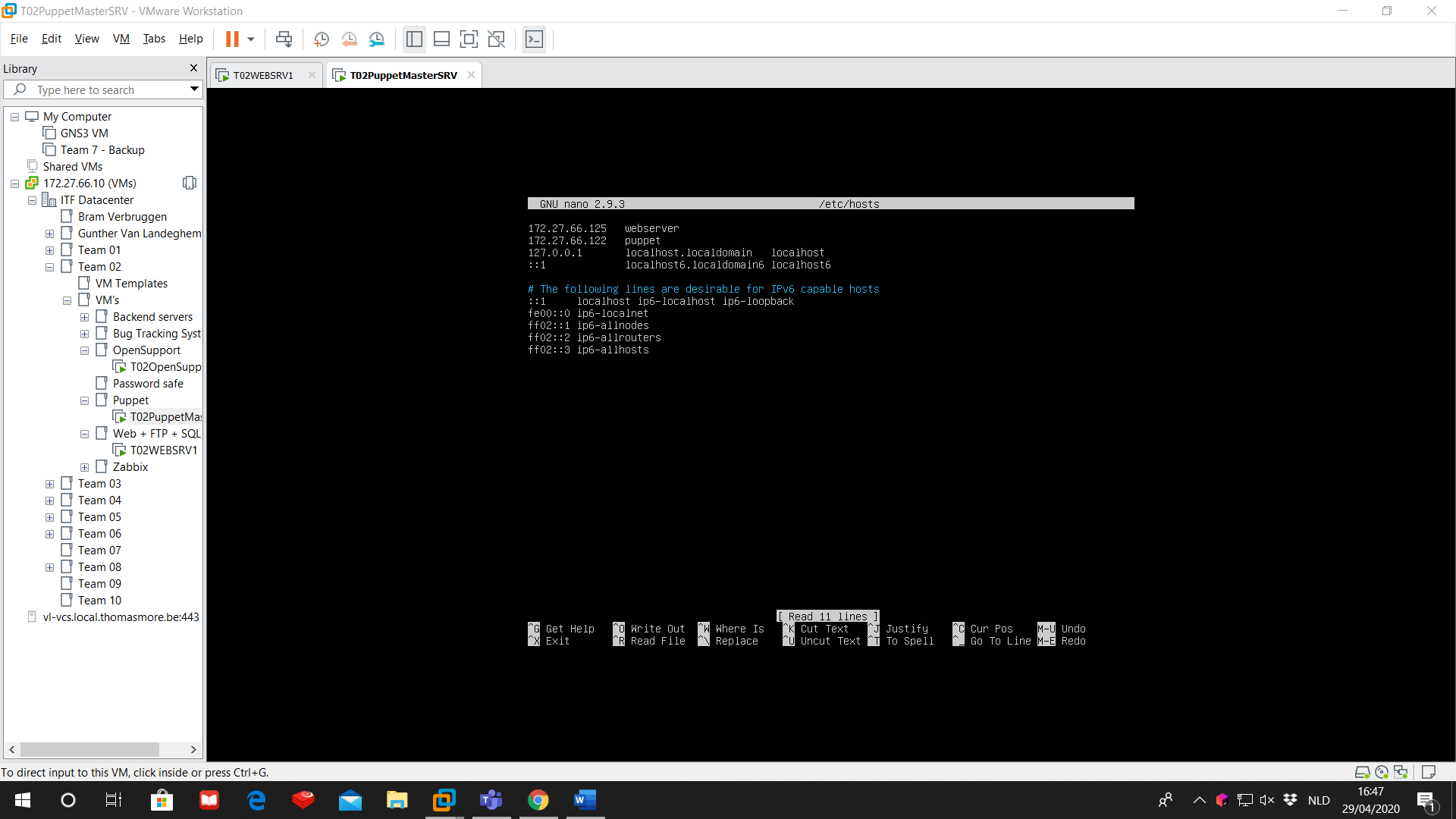
Again, we change /etc/cloud/cloud.cfg to preserve the new hostname.



We’ll change /etc/hostname to ‘webserver’.



And finally we’ll change /etc/hosts to add an entry for the webserver agent and an entry for the puppet master.



Also here, reboot the machine to save the new host settings.

* 1. NTP

We need to install NTP (Network Time Protocol) on both master and agents. We need NTP to correctly sign the certificates that are needed for an agent so it can work with it’s master.

sudo apt-get install ntp ntpdate

To synchronize with the NTP servers:

sudo ntpdate -u 0.ubuntu.pool.org

Change the timezone to Europe/Brussels

sudo timedatectl set-timezone Europe/Brussels

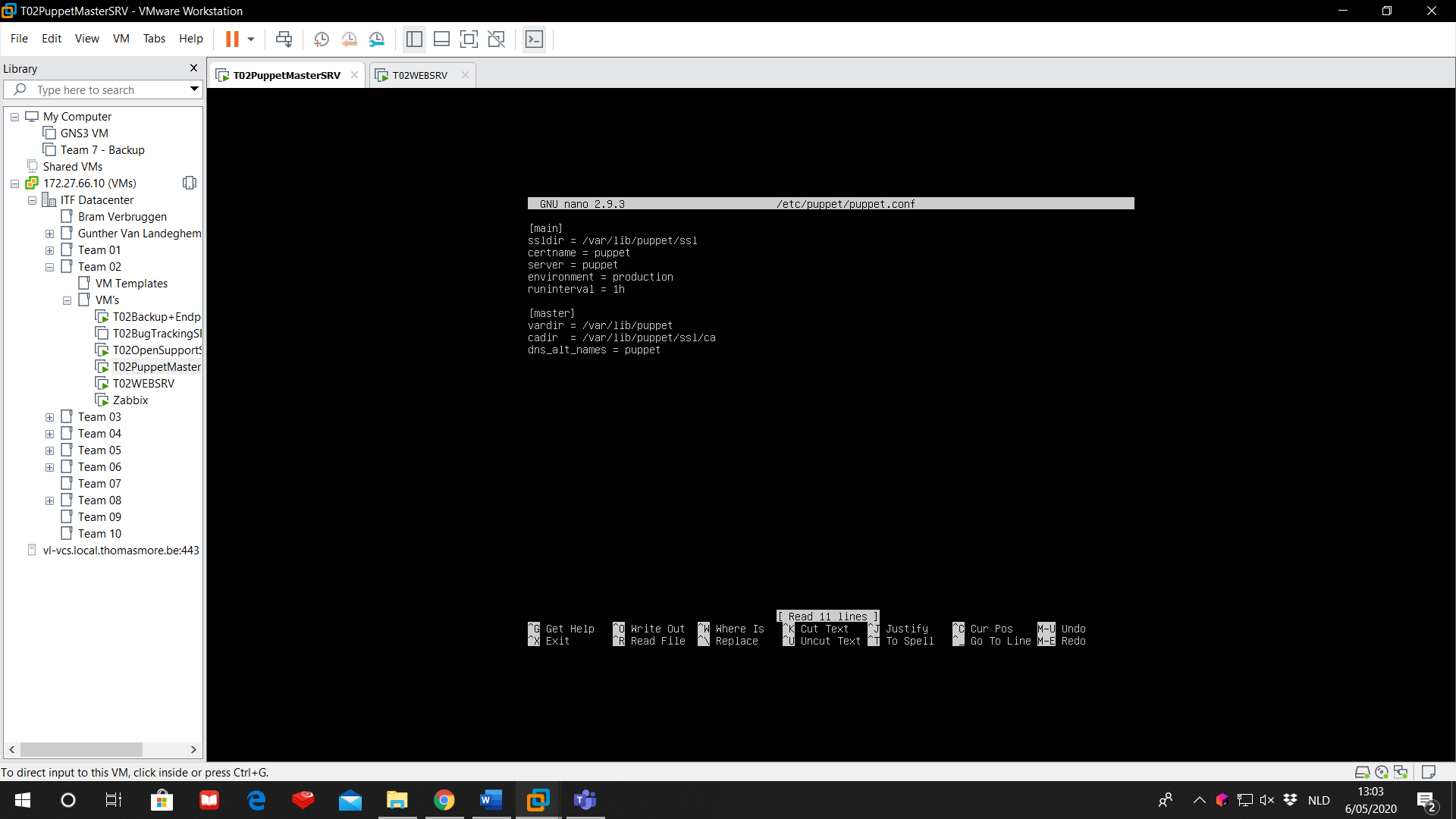
1. Installation
   1. Puppet server

Back to the puppet master, we download the package for puppet:

sudo apt-get install puppetmaster

Now we go to the configuration file in /etc/puppet/puppet.conf

This is how the configuration on the master should look like:



The config file has 4 types of modes:

[main]: the global (default) section, used by all services. If the same configuration also appears in another section, that configuration will override the one in here.

[master]: used by the Puppet master and the Pupper certificated authority

[agent]: used by the puppet agent

[user]: used by the ‘puppet apply’ command

On the master server, we only need [main] and [master].

Below is a short explanation for every parameter:

ssldir: where the Puppet master stores its certificates and private keys. This directory is has puppet as its owner and can’t be accessed.

certname: the certificate name, often the same as the hostname

server: the name of the puppet master to which agents request their configurations

environment: the environment to request

runinterval: every interval, the agent will check if the manifest on the master has changed and if it did, the agents will apply these changes in its configuration. The runinterval can be described in seconds (s), minutes (m), hours (h), days (d) of years (5).

vardir: where Puppet stores its dynamic and growing data.

cadir: the root directory of the certificate authority

dns\_alt\_names: a list of extra hostnames that the server can use to serve its agents.

Now that the master is configured we can start the Puppet master service:

sudo systemctl start puppet-master

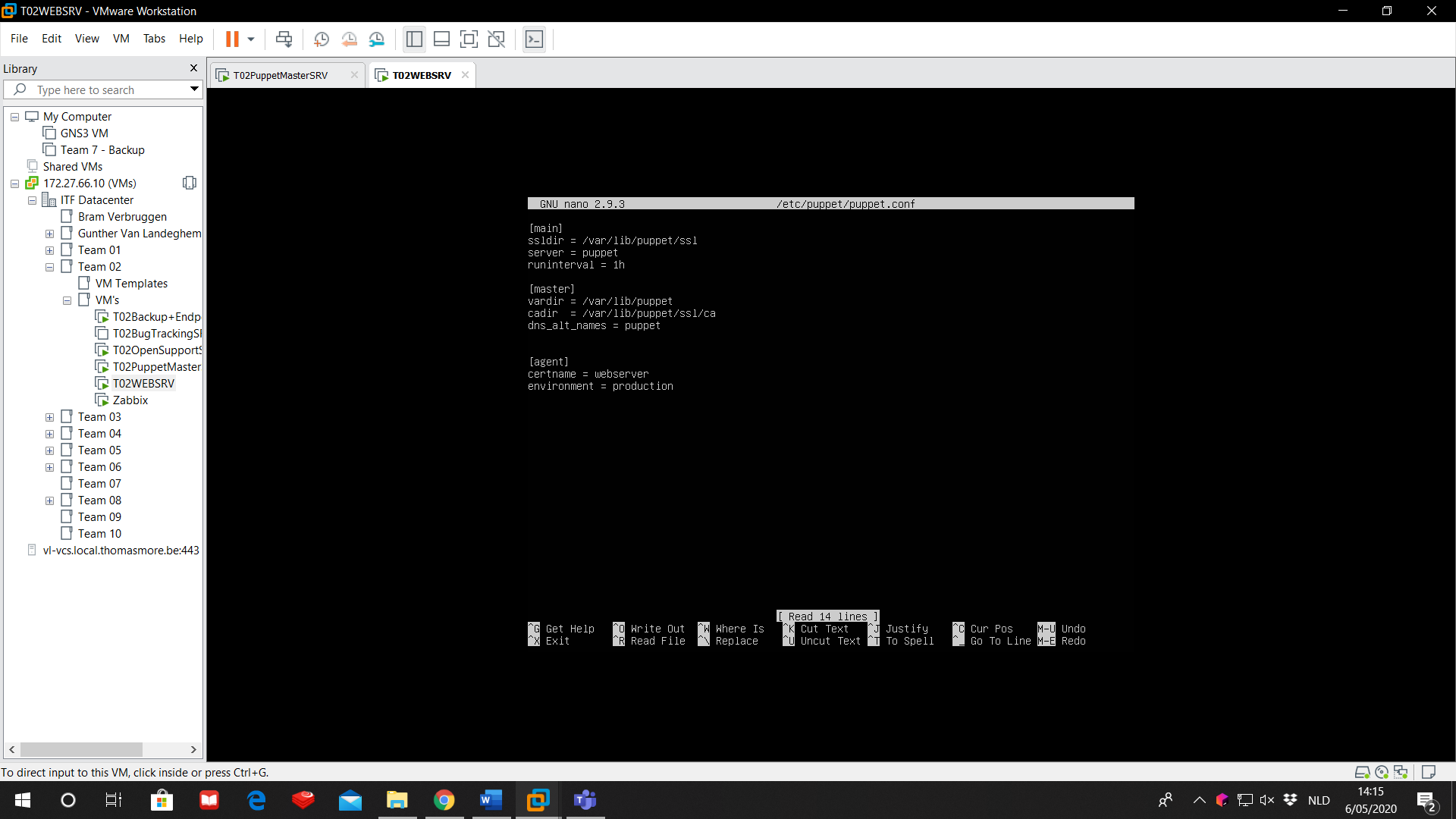
sudo systemctl enable puppet-master

* 1. Puppet agent

We install the puppet agent service:

sudo apt-get install puppet

Again, we go to the configuration file in /etc/puppet/puppet.conf and we change it to this screenshot.



Finally, we will start the agent and set it to automatically start while booting the machine.

sudo /usr/bin puppet resource service puppet ensure=running enable=true

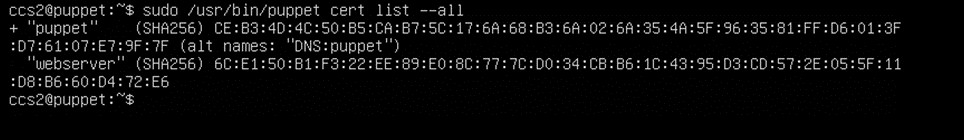
1. Signing certificates

On startup, the agent has sent a certificate to the master. This certificates will make it possible for the agent to retrieve the changes in configuration from the master.

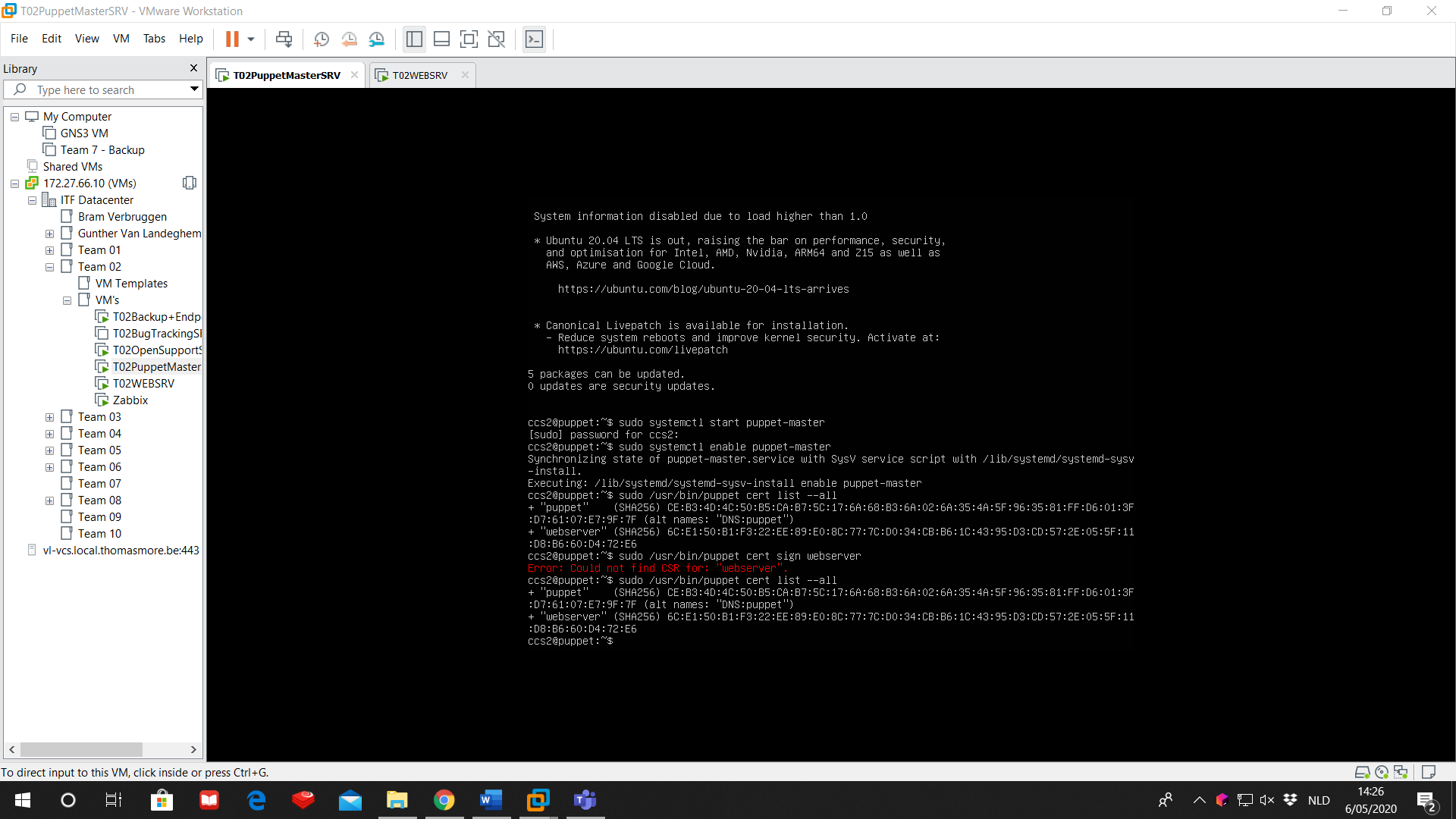
You can check if the master received this certificate by doing

sudo /usr/bin/puppet cert list –all

on the master. The output will look like this:

The “+” in front of the certificate shows if the certificate is already signed or not. As you can see, the certificated for the node with the name “webserver” hasn’t been signed yet. Do to this, we use

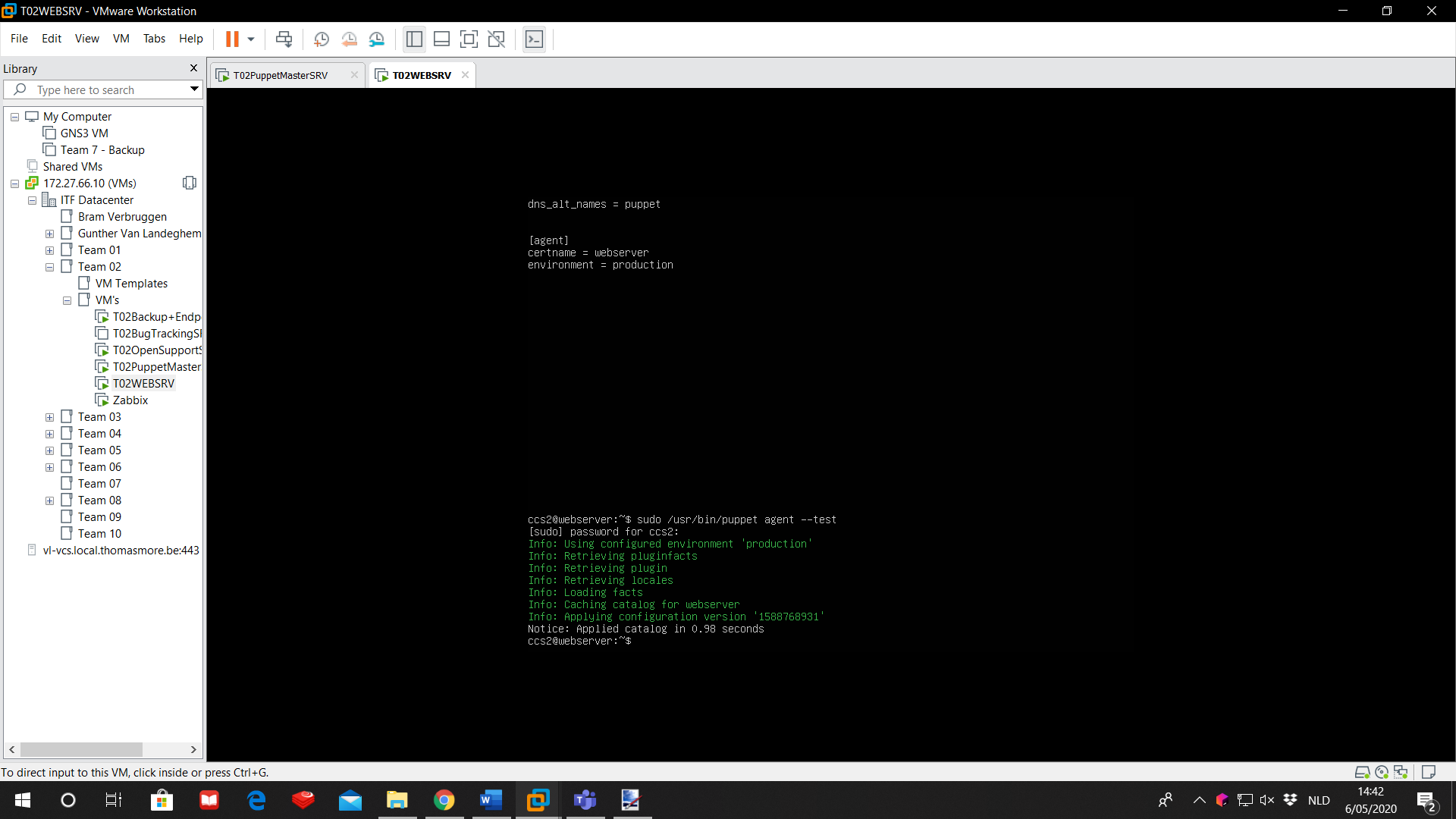
sudo /usr/bin/puppet cert sign webserver



Now we can see in the certificates list that the certificate for the node “webserver” has been signed.

We will check this as well on the agent side with:

sudo /usr/bin/puppet agent –test



If you see something similar like this appear, then your certificate is signed successful.

1. Installing Puppet modules

We are running and managing an apache webserver with Puppet. On the master we will create a manifest. This manifest will be sent to the designated agents. The agents will build their configuration file based on this manifest.

To create a manifest for based on a certain program, we’ll have to install a certain module. These modules can be found on Puppet Forge. There are many modules for many programs, so we can use Puppet to configure a wide variety of tools.

To get the module for apache from Puppet Forge we’ll enter on the master server:

sudo puppet module install –-environment=production puppetlabs-apache

You’ll notice that Puppet created a new directory:

/etc/puppet/code/

1. Creating the manifest file for apache

To start, we will create the location for our manifests. We are placing our manifests in

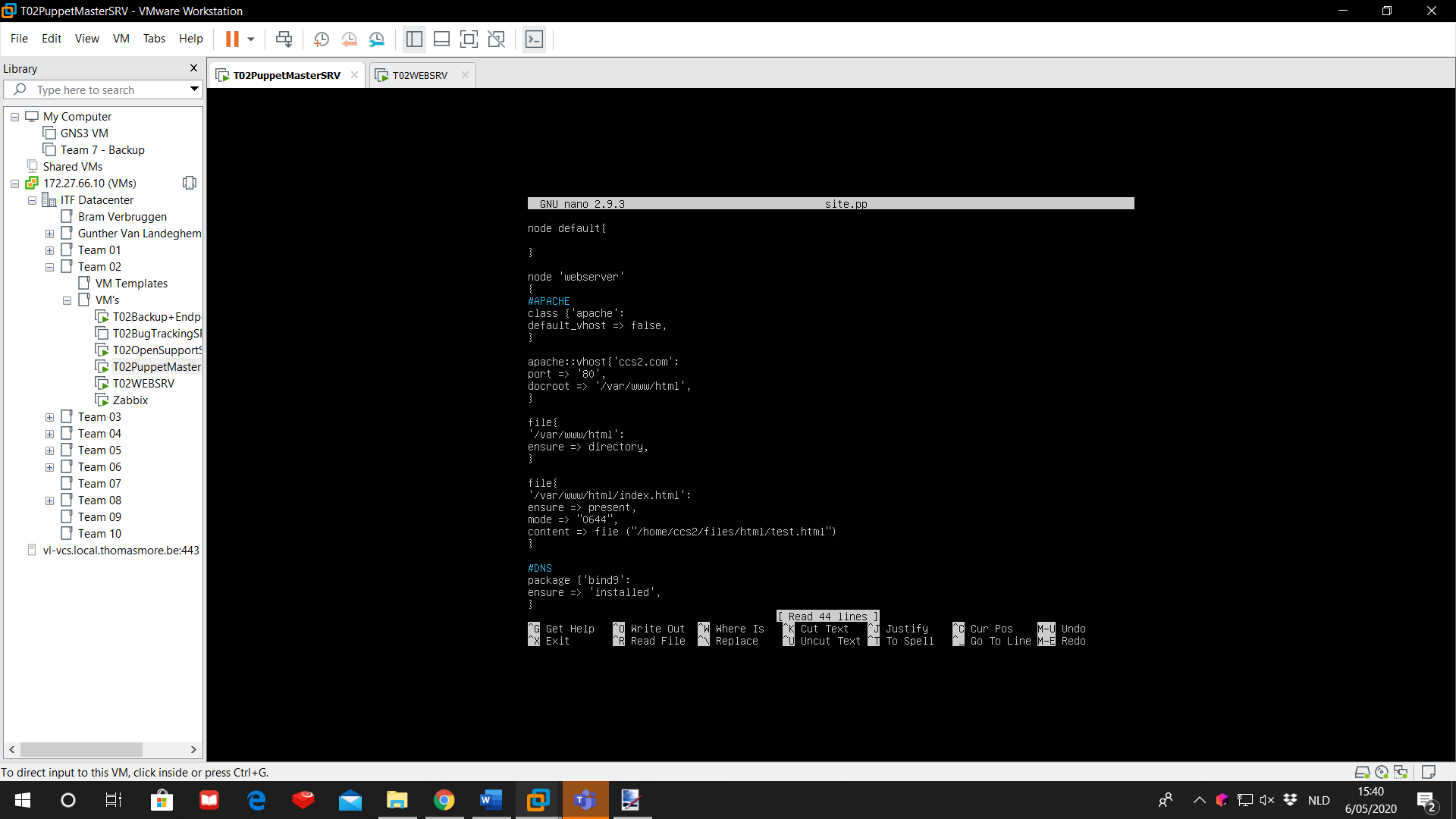
/etc/puppet/code/environment/production/manifests

You will probably have to create this directory structure first.

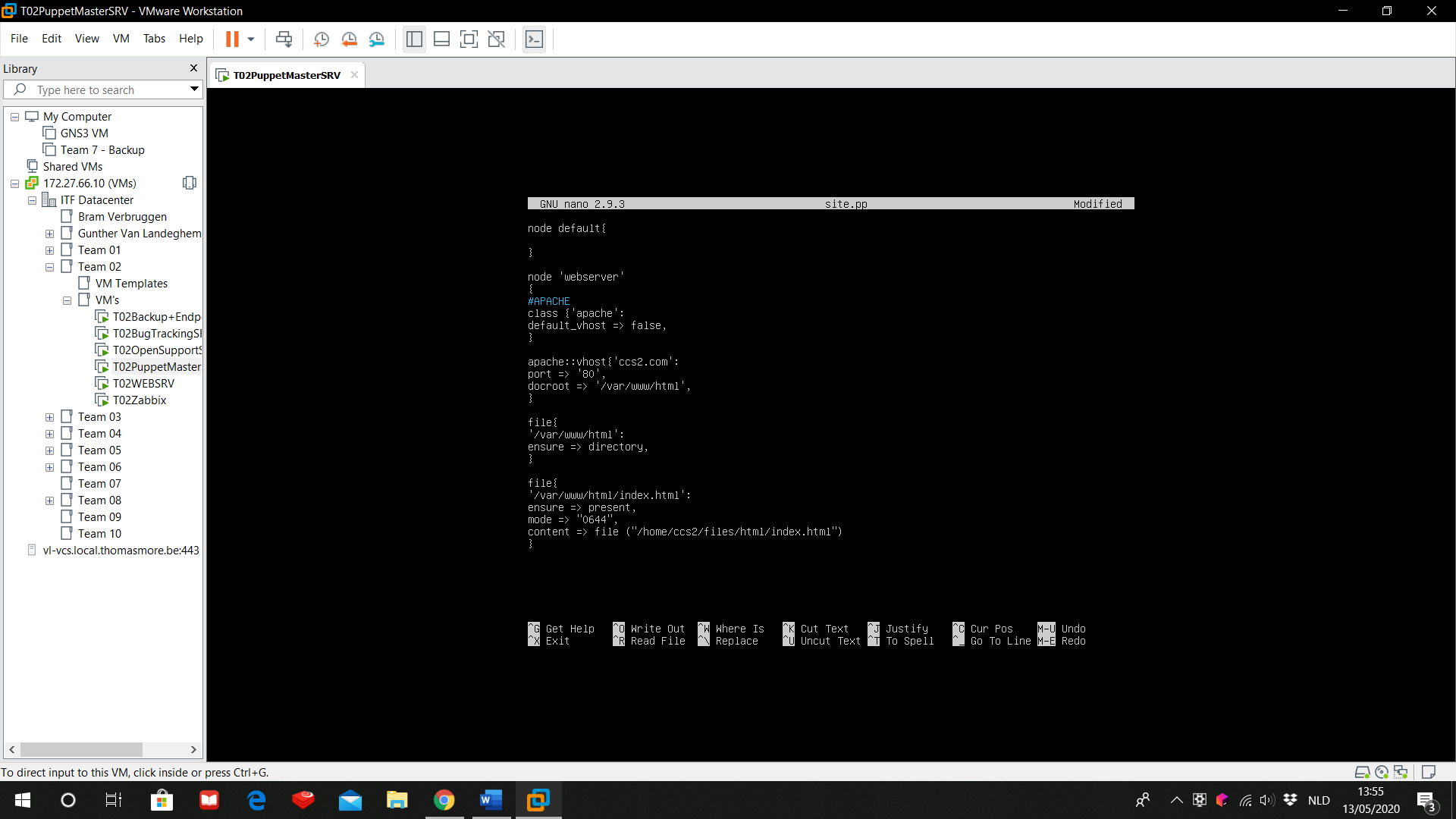
In this manifests directory, we create a manifest site.pp

sudo nano site.pp

The configuration has its own section, marked with {}. We begin with an empty default section.



Then we create a node for our webserver agent.



We need to write webserver between single quotes (‘’) because webserver is the hostname of our agent, while default is part of the Puppet syntax.

The first segment of the code:

class{‘apache’:

default\_vhost => false,}

Will install Apache on the agent without setting up a default Apache server. The default settings depend on your operating system and are not suggested in a production environment. We will create our own virtual host with:

apache::vhost{‘ccs2.com’:

port => ‘80’,

docroot => ‘/var/www/html’,}

Apache will set up this virtual host with name ‘ccs2.com’. The host will listen on port 80 and the default directory of this host will be /var/www/html.

Next we will make sure that the directory /var/www/html is available on the agent:

file{

‘/var/www/html’:

ensure => directory,}

With ‘ensure’ the agent will check if ‘/var/www/html’ is a directory that already has been created on the agent. If it isn’t, then the agent will create this directory.

We last part is similar, but with some extra additions:

file{

‘/var/www/html/index.html’:

ensure => present,

mode => “0644”,

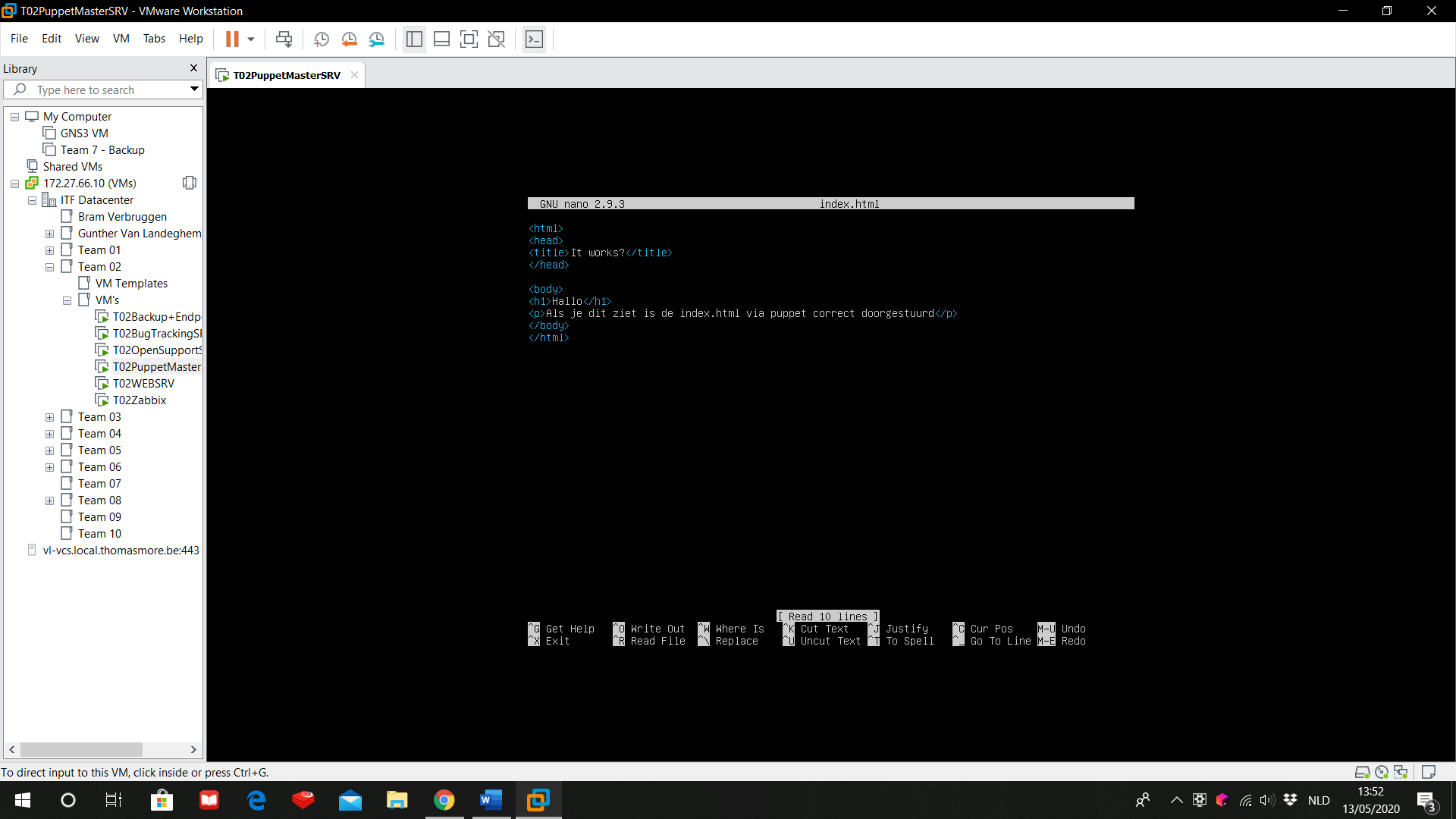
content => file{“/home/ccs2/files/html/index.html”)}

In this section, the agent will check if there is a file /var/www/html/index.html and will created it if there isn’t. The privileges of this file will be set to 0644. This works similar to chmod in Linux. “0644” will give reading and writing privileges for the user and only reading privileges for the group and others. For the content of index.html, we are referring to another file on the Puppet server. For our example, this is /home/ccs2/files/html/test.html. We will use this file to build the HTML-file on the agent.

* 1. Intermezzo: create the HTML file on the master

We will create a file on /home/ccs2/files/html/test.html. By default we are working already in /home/ccs2 so we have to create the directory ‘files/html’. In this directory we will create ‘test.html’.

To test if the agent correctly receives our configuration, we keep the content of the HTML-file very basic.



* 1. Using the correct manifest directory

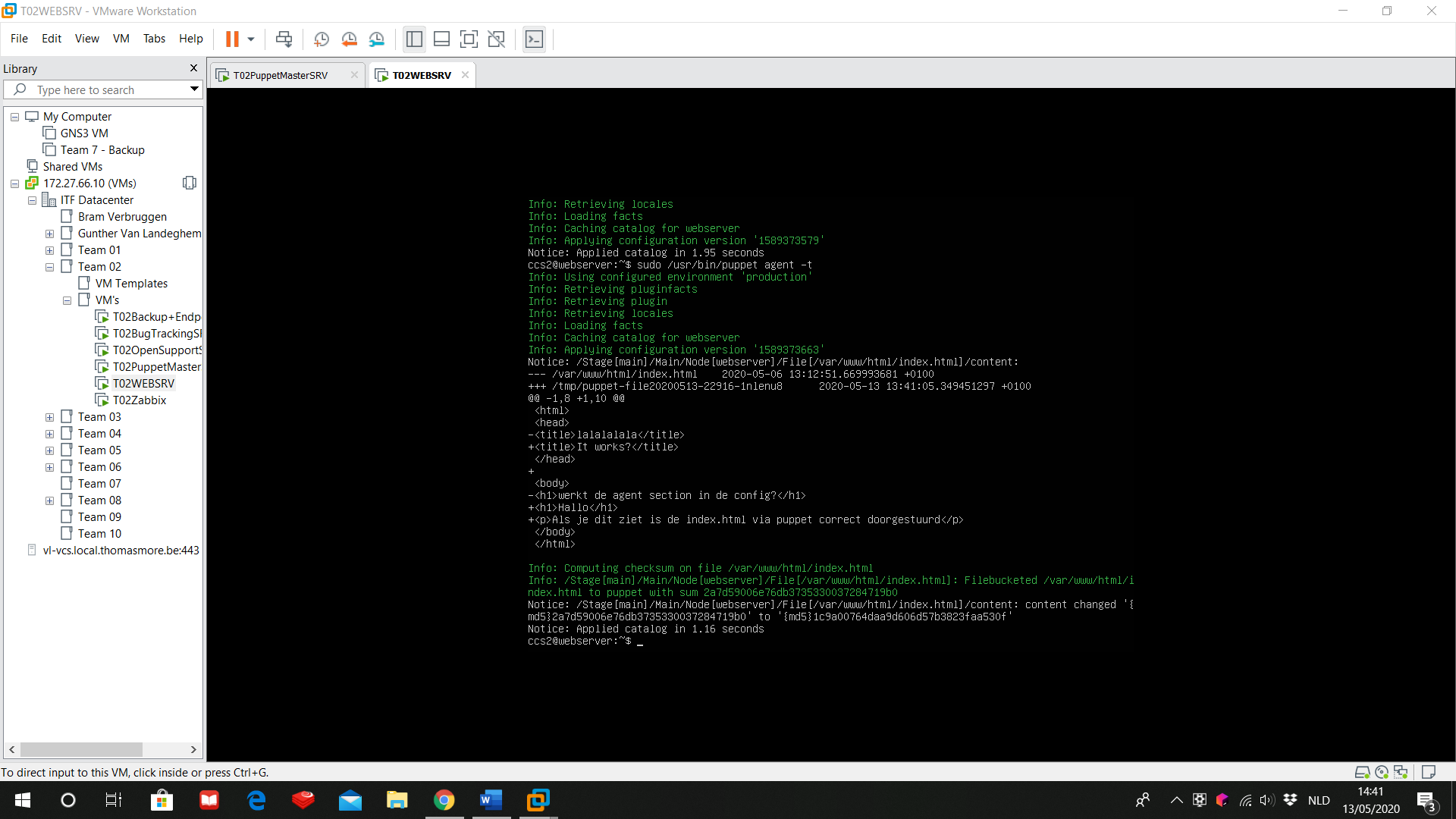
To make sure that the agent will look on the right place for the manifests, we do:

puppet apply /etc/puppet/code/environments/production/manifests

Finally, we will check if all our configurations are deployed correctly to the agent.

On the agent, we’ll enter:

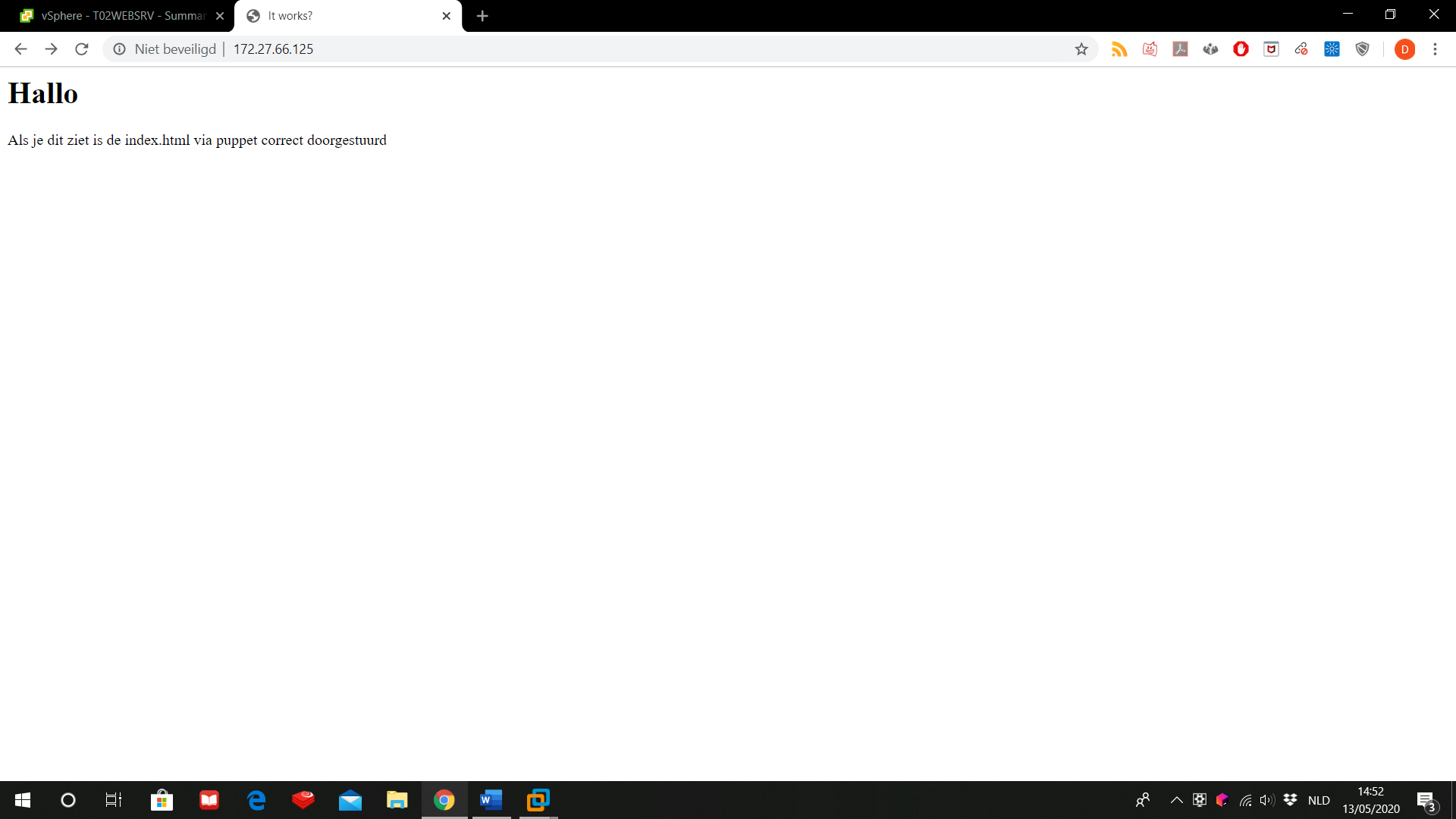
sudo /usr/bin/puppet agent -t



In the upper green text, the master will be contacted and the right manifests will be retrieved.  
The white text describes all the changes that have been made to the manifest in comparison with the current configurations on the agent.

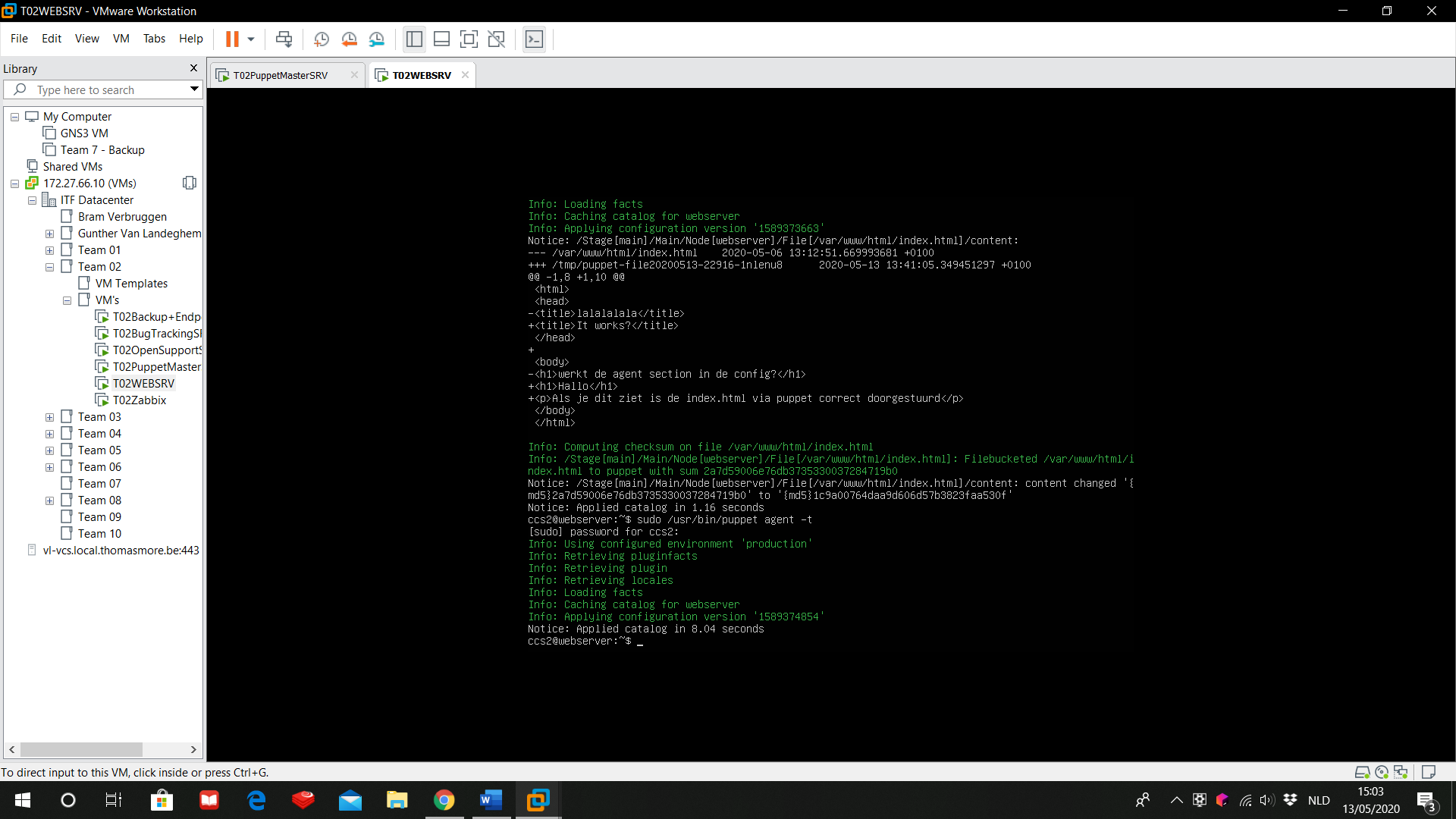
In the last bit of text, the agent will check if all the changes have been retrieved securely and without errors.

As a final visual check, we will browse to the IP of our webserver. If the HTML-file matches the text we wrote in section 5.1, then the configuration is successful.



1. Known issues
   1. No changes have been made to the agent

The first thing you’ll have to do is check if actual changes have been detected by the agents. To do this, you can just run /usr/bin/puppet agent -t again.



If the result looks like the above picture (no description of all the changes), it means that the agent hasn’t found any changes made in the manifest file.

* 1. Certificate error

Make sure you are using the right account to retrieve manifests from the master. Each user makes it’s own key-pair with the master. If you use another account than normal, it may cause a certificate mismatch.

Afbeelding met schermafbeelding

Automatisch gegenereerde beschrijving

We used puppet as root, so don’t forget to put sudo in front of the command.

Afbeelding met schermafbeelding

Automatisch gegenereerde beschrijving