Autor: Guillermo Bellettini

# Servidor Centralizado con  herramienta puppet

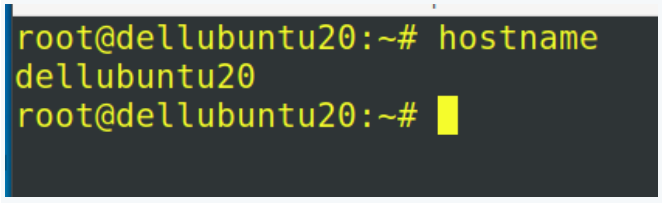
Puppet es una herramienta que permite gestionar, administrar y desplegar configuraciones personalizadas a otros Servidores llamados “Nodos Clientes/Agentes”.

## INSTALACIÓN DE PUPPET: NODO MAESTRO y NODO AGENTE/CLIENTE

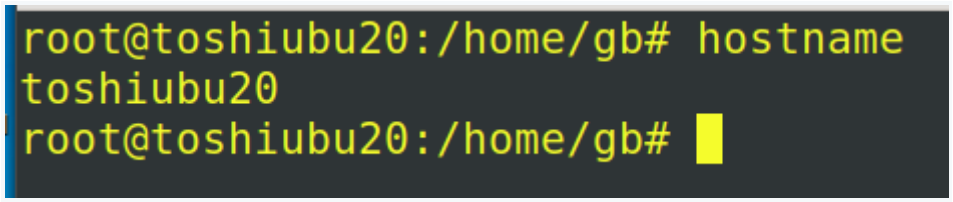
**Paso 1 – Configurar los servidores:**

1. Nos conectamos al servidor “maestro” y ejecutamos el siguiente comando para averiguar el nombre de máquina:

$ hostname




2. Ahora nos conectamos al “Nodo Cliente” y ejecutamos el mismo comando:




3. Ahora editamos el fichero /etc/hosts  en el Nodo Maestro:

$ nano /etc/hosts

Añadimos las siguientes líneas en el fichero /etc/hosts:
192.168.1.156 puppetmaster puppet dellubuntu20
192.168.1.200 puppetclient1 toshiubu20

Guardamos los cambios en el fichero y salimos.

**4.** Ahora editamos el fichero /etc/hosts  en el Nodo Cliente o Nodo Agente
$ nano /etc/hosts
Añadimos las siguientes líneas en el fichero /etc/hosts:

192.168.1.156 puppetmaster puppet dellubuntu20
192.168.1.200 puppetclient1 toshiubu20

Paso 2 – Instalamos Puppet Server (Master Node)
Nos conectamos al nodo Maestro

5. Descargamos el paquete debian que añadirá los repositorios puppet:

```
$ wget https://apt.puppetlabs.com/puppet7-release-focal.deb
```

Ahora ya podemos instalar dicho paquete para que nos añada los repositorios de Puppet

```
$ dpkg -i puppet7-release-focal.deb
```

6. Una vez añadido el repo PPA, debemos actualizar la lista de repositorios:

```
$ apt update
```

Procedemos a instalar el paquete "puppetserver"

```
$ apt install puppetserver -y
```

7. Ahora editamos el fichero de configuración de puppetserver:

```
$ nano /etc/default/puppetserver
```

Por defecto puppet, viene configurado para usar 2GB de memoria. En caso de no tener tanta memoria RAM disponible en nuestro server, podemos reducir a 1GB o cualquier otro valor, ej: 512m

```
JAVA_ARGS="-Xms512m -Xmx512m -Djruby.logger.class=com.puppetlabs.jruby_utils.jruby.Slf4jLogger"
```

Guardamos los cambios y salimos del fichero de puppetserver.

8. A continuación iniciamos el servicio de puppet en el nodo maestro y lo configuramos para que arranque cada vez que se reinicia el sistema:

```
$  systemctl start puppetserver
$  systemctl enable puppetserver
```

9. Verificamos que se ha arrancado correctamente:

```
$  systemctl status puppetserver
```

**Paso 3 – Instalar Puppet Agent (Agent Node)**

Nos debemos asegurar que hemos configurado correctamente el fichero /etc/hosts en este nodo agente tal como se indicó en el primer paso.

10. Descargamos el paquete debian que añadirá los repositorios puppet:

```
$ wget https://apt.puppetlabs.com/puppet7-release-focal.deb
```

Ahora ya podemos instalar dicho paquete para que nos añada los repositorios de Puppet

```
$ dpkg -i puppet7-release-focal.deb
```

11.  Una vez añadido el repo PPA, debemos actualizar la lista de repositorios:

```
$  apt update
```

Procedemos a instalar el paquete "puppet-agent"

```
$  apt install puppet-agent -y
```

12. Cuando haya finalizado la instalación del paquete "puppet-agent", editamos el fichero de configuración:

```
$  nano /etc/puppetlabs/puppet/puppet.conf
```

Añadimos las siguientes líneas para definir los detalles del NODO master:

```
[main]
```

certname = toshiubu20
server = dellubuntu20

Guardamos cambios y salimos del fichero.

13. A continuación, iniciamos el servicio de puppet y lo configuramos para que se inicie automatiamente cada vez que el sistema operativos se reinicie:
$ systemctl start puppet
$ systemctl enable puppet

14. Once done, verify the Puppet agent service is running properly:
$ systemctl status puppet

**Paso 4 – Firmar los certificados de los Nodos Agentes**

15. Nos logueamos en el Nodo maestro y ejecutamos el siguiente comando para listar los certificados disponibles:
$ /opt/puppetlabs/bin/puppetserver ca list --all

```
root@dellubuntu20:~# /opt/puppetlabs/bin/puppetserver ca list --all
Signed Certificates:
    toshiubu20         (SHA256)  18:76:85:D5:24:A8:75:8D:89:4C:78:37:2F:1C:B2:40:12:D0:73:03:C5:77:62:CE:28:B4:87:86:7D:6B
:75:E0  alt names: ["DNS:toshiubu20"]
    dellubuntu20       (SHA256)  BC:2A:93:BC:F4:BA:9B:26:EC:B1:54:E2:27:A0:74:EC:97:32:29:98:94:AC:9A:D1:9C:59:3E:30:B0:62
:ED:E4  alt names: ["DNS:puppet", "DNS:dellubuntu20"]   authorization extensions: [pp_cli_auth: true]
root@dellubuntu20:~#
```

NOTA: nos tiene que aparecer tanto el certificado del nodo agente (toshiubu20) como el certificado del propio Nodo Maestro (dellubuntu20). Si no vemos el certificado del nodo agente, podemos ir al **nodo agente** y ejecutar el siguiente comando para que genere un certificado y una petición de firma:
$ /opt/puppetlabs/bin/puppet agent --test

16. Procedemos a firmar los certificados en el Nodo Maestro:
$ /opt/puppetlabs/bin/puppetserver ca sign --all

17. Ahora vamos al nodo agente y probamos la conexión del nodo agente contra el nodo Maestro.

$ /opt/puppetlabs/bin/puppet agent --test => es el mismo comando con el que también se generan certificados y peticiones de firma

## Ejemplo de Despliegue de un servidor LAMP (Linux, Apache, MySql, Php )

En el nodos Maestro, posicionarse en el directorio de los módulos de Puppet:

$ cd /etc/puppetlabs/code/modules/

Ahora crear los siguientes directorios:

```
$ mkdir lamp
$ cd lamp
```

y dentro del directorio “lamp” crea un nuevo directorio con el nombre “manifests”:

```
$ mkdir manifests
```

```
root@dellubuntu20:/etc/puppetlabs/code/modules/lamp/manifests# pwd
/etc/puppetlabs/code/modules/lamp/manifests
root@dellubuntu20:/etc/puppetlabs/code/modules/lamp/manifests#
```

dentro del directorio “/etc/puppetlabs/code/modules/lamp/manifests”, crear el fichero **init.pp** y añadirle el siguiente código:

```
class lamp {
 # execute 'apt-get update'
 exec { 'apt-update':                    # exec resource named 'apt-update'
   command => '/usr/bin/apt-get update'  # command this resource will run
 }

 # install apache2 package
 package { 'apache2':
   require => Exec['apt-update'],       # require 'apt-update' before installing
   ensure => installed,
 }

 # ensure apache2 service is running
 service { 'apache2':
   ensure => running,
 }

 # install mysql-server package
 package { 'mysql-server':
   require => Exec['apt-update'],       # require 'apt-update' before installing
   ensure => installed,
 }

 # ensure mysql service is running
 service { 'mysql':
   ensure => running,
 }

 # install php package
 package { 'php':
```

```
  require => Exec['apt-update'],       # require 'apt-update' before installing
  ensure => installed,
 }
 # ensure info.php file exists
 file { '/var/www/html/info.php':
  ensure => file,
  content => '',    # phpinfo code
  require => Package['apache2'],       # require 'apache2' package before creating
 }
}
```

```
root@dellubuntu20:/etc/puppetlabs/code/modules/lamp/manifests# ls -l
total 4,0K
-rw-r--r-- 1 root root 1,1K feb 12 18:12 init.pp
root@dellubuntu20:/etc/puppetlabs/code/modules/lamp/manifests#
```

## Crear el fichero de manifiesto principal

Un manifiesto es un fichero que contiene configuraciones de cliente y se usa para poder instalar, gestionar y administrar nodos clientes. El fichero manifiesto principal de Puppet está en la carpeta: /etc/puppetlabs/code/environments/production/manifests

```
root@dellubuntu20:/etc/puppetlabs/code/environments/production/manifests# ls -l
total 0
root@dellubuntu20:/etc/puppetlabs/code/environments/production/manifests# nano site.pp
```

$ nano /etc/puppetlabs/code/environments/production/manifests/site.pp

Añadir las siguientes líneas:

```
node default { }

node 'toshiubu20' {
 include lamp
}
```

Guardar los cambios y salir del fichero.

Algo por hacer en el NODO Agente, si ya teníamos instalado “apache2”..... Es necesario ir a la siguiente carpeta para añadir código fuente a la página /var/www/html/info.php

$ nano /var/www/html/info.php

añadimos la siguiente línea:

```
<?php phpinfo(); ?>
```

Guardamos cambios en el fichero y salimos:

Para poder desplegar esta configuración desde el NODO MAESTRO hacia el NODO AGENTE, debemos entrar en el nodo agente y lanzar el siguiente comando para que se conecte al servidor maestro y se ejecute el despliegue.

En la consola del nodo agente ejecutamos:

```
$ /opt/puppetlabs/bin/puppet agent --test
```

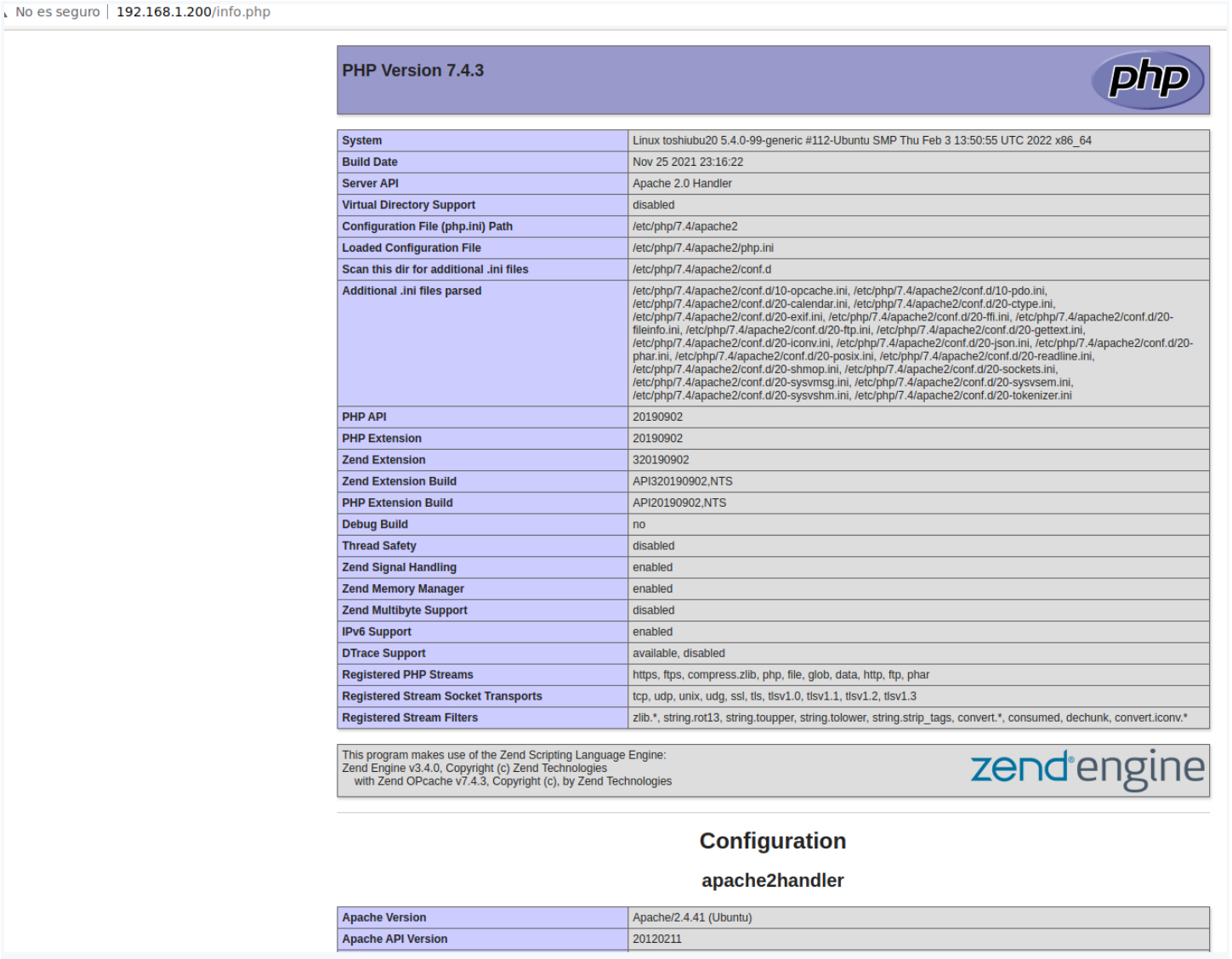
nos debería devolver una respuesta parecida a esto:

```
Info: Using configured environment 'production'
Info: Retrieving pluginfacts
Info: Retrieving plugin
Info: Retrieving locales
Info: Caching catalog for puppet-agent
Info: Applying configuration version '1594188800'
Notice: /Stage[main]/Lamp/Exec[apt-update]/returns: executed successfully
Notice: /Stage[main]/Lamp/Package[apache2]/ensure: created
Notice: /Stage[main]/Lamp/Package[mysql-server]/ensure: created
Notice: /Stage[main]/Lamp/Package[php]/ensure: created
Notice: /Stage[main]/Lamp/File[/var/www/html/info.php]/ensure: defined content as '{md5}d9c0c977ee96604e48b81d795236619a'
Notice: Applied catalog in 73.09 seconds
```

Para comprobar que se ha instalado el server LAMP, abrimos un navegador y escribimos:

http://agent-ip-address/info.php.

en mi caso: http://192.168.1.200/info.php


No es seguro | 192.168.1.200/info.php
PHP Version 7.4.3
php
System
Linux toshiubu20 5.4.0-99-generic #112-Ubuntu SMP Thu Feb 3 13:50:55 UTC 2022 x86_64
Build Date
Nov 25 2021 23:16:22
Server API
Apache 2.0 Handler
Virtual Directory Support
disabled
Configuration File (php.ini) Path
/etc/php/7.4/apache2
Loaded Configuration File
/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files
/etc/php/7.4/apache2/conf.d
Additional .ini files parsed
/etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini,
/etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini,
/etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-
fileinfo.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini,
/etc/php/7.4/apache2/conf.d/20-iconv.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-
phar.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini,
/etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini,
/etc/php/7.4/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvsem.ini,
/etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini
PHP API
20190902
PHP Extension
20190902
Zend Extension
320190902
Zend Extension Build
API320190902,NTS
PHP Extension Build
API20190902,NTS
Debug Build
no
Thread Safety
disabled
Zend Signal Handling
enabled
Zend Memory Manager
enabled
Zend Multibyte Support
disabled
IPv6 Support
enabled
DTrace Support
available, disabled
Registered PHP Streams
https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports
tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters
zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, convert.iconv.*
This program makes use of the Zend Scripting Language Engine:
Zend Engine v3.4.0, Copyright (c) Zend Technologies
with Zend OPcache v7.4.3, Copyright (c), by Zend Technologies
zend engine
Configuration
apache2handler
Apache Version
Apache/2.4.41 (Ubuntu)
Apache API Version
20120211