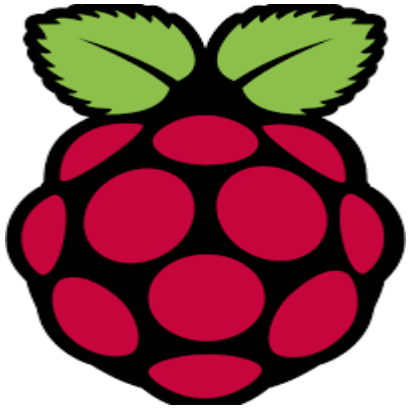

Docker Swarm Cluster con Raspberry Pi. Parte 1

Descripción

Rapsberry Pi Setup

A veces nos cohibimos de probar ciertas tecnologías que necesitan más de una Pc porque no poseemos varias PC para probarla por el costo prohibitivo de estas, pero y te dijera que es mucho mas barato usando mini computadoras o «Single Board Computers» (SBC) como las Raspbery Pi, Orange Pi, Banana Pi, etc.

Este tutorial está enfocado en lo que dispongo:

– 1 x Raspberry Pi 3B+
– 2 x Orange Pi Prime (muy parecida a la Raspberry Pi 3B+ pero con un poco más de recursos)

Preparar las tarjetas micro SD y flashear imágenes

En nuestro caso como que no usaremos interfaz gráfica usaremos en el caso de Raspbian la versión lite

Y eso mismo debemos hacer con las Orange Pi Prime, en nuestro caso usamos Armbian la versión de consola y basada en Debian Buster 10.x

Una vez que tengamos las imágenes recomiendo usa el software Etcher, aunque si están en Ubuntu linux con la utilidad de discos pueden restaurar la imagen a la tarjeta micro SD *(uSD de ahora en adelante para abreviar)* sin ningún tipo de problemas.

Sabías que hay una variante muy simple de acceder físicamente a las SBC sin usar display?

A veces es engorroso tener que estar usando un TV o display y cambiar de un SBC a otro con teclado/mouse USB para configurar algo, a veces ni siquiera tenemos un display disponible con HDMI o un adaptador HDMI > VGA…

La variante es la consola serial, todo lo que necesitas es un adaptador de USB a Serie que los puedes encontrar en internet por menos de 2 USD (menos shipping) y en Cuba date una vuelta por el grupo de Telegram Arduino Cuba y pregunta por Vladimir

Si buscas el pinout *(mapa de pines y sus funciones)* de las SBC encontrarás que tienen dos pines nombrados RX y TX, pues estos corresponden al puerto serial.

En Armbian están habilitadas por defecto y casi todas las boards Orange Pi tienen un conector de 3 pines dedicado a estos fines.

En el caso de Raspbian hay que habilitar un detalle en la configuración que veremos a continuación.

En ambos cualquier caso puedes usar un software de terminar serial para conectarte, en Windows recomiendo el venerable Putty, en linux tenemos la mejor opción que es `minicom`

Que necesitan para comunicarse:

- Que puerto serie usa el adaptador
- La velocidad de comunicación del puerto

En el primer caso para Windows deben determinar en el administrador de dispositivos, en linux basta con hacer un `tail -f /var/log/syslog | grep tty` mientras conectas el dispositivo y verás algo como:

```
Jun 14 00:10:41 agatha2 kernel: [53182.056895] usb 1-1: ch341-uart converter now attached to ttyUSB0
```

En este caso el puerto serie será `/dev/ttyUSB0` otros tipos de adaptadores generár otros dispositivos como /dev/ttyAMA0, /dev/ttyACM0, etc.

En el caso de la velocidad del puerto serie siempre se usa **115200 baudios**.

En el caso de linux luego de instalar minicom podemos ejecutar:

```
minicom --color=on -b 115200 -D /dev/ttyUSB0
```

Esto abrirá la terminal con soporte para colores y en el puerto que se muestra

Como configuro el Raspbian para que inicie con la consola en el puerto serie y no usar monitor

Simple, conecte la uSD a su PC/laptop y busque una partición/torre nombrada `boot` en esta abra un fichero llamado config.txt y al final hay una sección llamada **[all]**, debajo de ella escriba:

```
enable_uart=1
```

Salve el fichero, inserte la uSD en la Raspberry Pi, conecte el pierto serie a los pines adecuados, abra el software de terminal y alimente el Raspberry Pi, luego de un momento verá el inicio en la pantalla como si se tratara de un termina en un display.

Nota: *Putty tiene problemas para monstrar cosas con ncurses, así que evite usar herramientas que usan ncurses en la consola serial, por ejemplo el gestor de ficheros mc, en linux con minimcom no es problema.*

Iniciando la configuración

En Raspbian para Raspberry Pi las credenciales por defecto son

- user: pi
- password: raspberry

Lo primero que deben hacer cuando te logueas es cambiar el password y habilitar el acceso de root y el SSH, para cambiar el password de user pi simplemente digite lo siguiente y siga las instrucciones

```
passwd
```

Para cambiar el password de root es lo mismo pero con un detalle:

```
sudo bash
passwd root
```

De ahora en adelante asumiré que haz hecho `sudo bash` y estás trabajando como root.

Habilitando acceso por SSH

En raspbian es tan simple como crear un fichero nombrado **ssh** (en minúsculas) en la partición **boot**, incluso puedes hacerlo desde tu PC cuando formateas la uSD, si ya estas con el Raspbian iniciado puedes correr esto:

```
touch /boot/ssh
reboot
```

Ten en cuenta que el comando reboot reiniciará el sistema.

Dedicar casi toda la RAM a procesamiento y no a gráficos

Por defecto Raspbian dedica 128 MB de RAM al procesamiento de video y puede escalar a más bajo demanda, en este caso no usaremos video, así que le diremos que use el mínimo posible de RAM para VRAM, es simple edita desde tu PC el fichero que está en la partición **boot** llamado **config.txt** que ya posiblemente has modificado y escribir esto debajo de la sección llamada **[all]**

```
gpu_mem=16
```

Con esto le estas dando el mínimo de RAM al video y evitas que escale a mayores cantidades.

Que hora es?

Uno de los detalles de las SBC es que muy pocas tienen un Reloj de Tiempo Real (RTC) y pierden la fecha/hora en cada reinicio, si tienes internet cuando usas tu dispositivo o un server NTP en al red hay una manera simple de hacer que este sincronice y adquiera la hora en cada reinicio.

Lo primero es configurar la zona horaria, con la herramienta raspi-config y sus menues puedes lograrlo (usa ncurses, no recomendado si estás en consola serial) pero hay otra vía más directa y vieja:

```
sudo dpkg-reconfigure tz-data
```

En Cuba usamos «America/Havana»

Luego edita el fichero soguiente con el comando nano

```
nano /etc/systemd/timesyncd.conf
```

Luego llena la linea correspondiente a NTP con un server NTP en tu red o un server público

Por ejemplo:

```
NTP=0.pool.ntp.org 1.pool.ntp.org 2.pool.ntp.org 3.pool.ntp.org
```

Esto aunque todavía tu SBC no esté conectado a ninguna red, más de eso en brevel; aplicamos los cambios con los comandos:

```
systemctl daemon-reload
systemctl restart systemd-timesyncd
```

Procedemos a poner manualmente la hora, es tan simple como eso:

```
date -s "June 14 2020 00:35:30"
```

Por supuesto debes ajustar los valores, siempre usa hora militar para evitar problemas

Y si tengo un RTC de hardware para ponerle al Raspberry Pi

Pues felicidades, yo también tengo uno y los más comunes son los basados en los chips ds1307 y ds3231, consulta los tutoriales en internet para saber como conectarlos (hay unos que se venden listo para poner en el GPIO del Raspberry) y usa mi referencia para configurarlo, créeme te ahorraras dolores de cabeza.

Con las versiones de Rapsberry a partir de finales de 2019 habilitar un RTC es tan simple como añadir dos líneas al archivo `/boot/config.txt` al final de este:

```
dtparam=i2c_arm=on,i2c_arm_baudrate=400000
dtoverlay=i2c-rtc,ds3231
```

En el caso que tu RTC use un ds1307 pues sustituye el nombre del dispositivo y listo

Ojo, verifica que no hay otra línea casi al final del archivo que empiece con **dtparam=i2c_arm=on** si es así debes comentarla para que entre la nuestra a funcionar

Reinicia y listo, solo queda un paso, poner en hora el RTC que haz conectado, con el siguiente comando puedes inicialmente consultar la

hora del RTC paraverificar su funcionamiento y con el segundo aplicas la hora del sistema al RTC.

```
hwclock -r
hwclock -w
```

La prueba final es aparar la Raspberry y desconectarla de la corriente, luego de 2-3 minutos desconectada cuando inicia debe conservar la hora

Puedes en este punto instalar un server NTP para que sirva a las demás SBC de hora, es simple, lo veremos luego de comentar como obtener conectividad en la Raspberry

Conectándonos a una red con o sin Intenret

Para conectarlos la variante más simple es usar ethernet pues la configuración del Raspberry espera un server DHCP en la red ethernet para conectarse, simplemente conecta un cable erhernet al router de nauta hogar u otra red con DHCP y listo ya tienes conexión

Si no tienes nauta hogar puedes usar los datos móviles, conecta tu cell por USB y comparte la conexión vía USB en los ajustes del cell, a esto se conoce como «thethering» simplemente abre los datos móviles y listo.

Tip: si te preocupa el consumo de datos, revisa este gist con detalles que te harán la vida fácil en este aspecto.

Si te vas a conectar inicialmente por Wifi con el comando `raspi-config` en el apartado de red puedes darle los datos y te conectará a esa red WiFi.

Si estas en una rede de la universidad/trabajo/etc y usas proxy nada mas fácil que usar el proxy para todo lo que sigue, en este caso debes exportar dos variables de entorno en la consola como sigue:

```
export http_proxy="http://user:password@ip.del.proxy.o.nombre:puerto/"
export https_proxy="http://user:password@ip.del.proxy.o.nombre:puerto/"
```

Ya te deben funcionar apt, wget, curl y hasta etcereta, por lo que se requiere un update y upgrade del sistema operativo, preparate pues estas a punto de descargar aprox 100MB de internet

```
apt get update && apt upgrade -y
```

Raspberry Pi como master del Cluster y como servidor «decente»

En mi diseño uso el Raspberry como master del cluster, como server de tiempo (es el único que tiene RTC) y como hotspot ya que al carecer de Switch ethernet pues tengo que morir con la red vía WiFi.

Pero sucede que Raspbian como SO hace sus sacrificios y simplificaciones que un sysadmin encontrará molestos, porque no es lo común o lo esperado en el apartado de red.

Veamos como hacer que Raspbian se porte de verdad como un server.

DHCPCD sucks: Sustituir dhcpcd x NetworkManager (nm)

Para ser un gestor de conectividad básico dhcpcd está ok, pero cuando le pide algo más de lo común falla miserablemente y crea complicaciones innecesarias, para sustituirlo simplemente haga lo siguiente:

```
apt install network-manager
systemctl stop dhcpcd
systemctl disable dhcpcd
systemctl mask dhcpcd
reboot
```

Esto reiniciará el sistema y ya tendremos a nm como gestor de red. Si no lo has hecho deberías aprender al menos lo básico de nmcli, pues está en el 90% de los linux actuales y entre ella y netplan dominarán el mercado en muy poco tiempo, algunos comandos simples para hacer cosa básicas:

- Listar dispositivos: *nmcli d s*
- Listar redes wifi: *nmcli d w*

- Conectarse a una WiFi: *nmcli device wifi connect «BSSID» password «P4s5w0rd__»*

Más configuraciones de la red

En nuestro caso usaremos la wifi como hostspot para el uso del cluster, por lo que debemos poner una IP fija en esta y configurar el hostspot el DHCP, el DNS, etc.

En nuestra config el servicio por defecto de systemd para la resolución de nombres (DNS) entra en conflicto por lo que lo deshabilitamos

```
systemctl stop systemd-resolved
systemctl disable systemd-resolved
systemctl mask systemd-resolved
```

IP fija en la Wifi

Es simple, lo hacemos por la vía tradicional y como nm es más inteligente que dhcpcd sabrá que es manualmente configurada, debemos editar el fichero correspondiente con el comando `nano /etc/network/interfaces` y añadir al final esto:

```
auto wlan0
iface wlan0 inet static
    address 10.42.1.1
    netmask 255.255.255.0
    dns-domain co7wt.cu
```

Salimos con F2 y verificamos que nm ya reconoce el interfaz como configurada manualmente «unamanaged»:

```
nmcli d s
[...]
wlan0 wifi unmanaged --
[...]
```

Instalar dnsmasq

Este será nuestro server DHCP/DNS en la red

```
apt install dnsmask
```

Ahora editamos el fichero en default con el comando `nano /etc/default/dnsmasq` y verificamos dos cosas:

- «ENABLED=1»
- «IGNORE_RESOLVCONF=yes»

Salvamos con F2 y salimos, ahora editaremos las config del server DNS y DHCP, salvamos la config por defecto que sirve de referencia porque tiene muchos comentarios importantes con el comando `mv /etc/dnsmasq.conf dnsmasq.conf_original`

Editamos el nuevo fichero de config con el comando `nano /etc/dnsmasq.conf` y pegamos el contenido siguiente, ajustando el dominio, el segmento de red, el DNS y el server de tiempo de la red a usarse.

```
# interface de escucha
interface=wlan0

# dominio local
domain=co7wt.cu

# rango
dhcp-range=10.42.1.1,10.42.1.200,255.255.255.0,24h

# asignación estática
dhcp-host=3a:bc:48:6b:41:8b,rpi0,10.42.1.1,24
dhcp-host=0c:8c:24:2c:50:be,opi1,10.42.1.2,24
dhcp-host=3a:bc:48:6b:41:8c,opi2,10.42.1.3,24
dhcp-host=3c:95:09:55:33:a7,agatha,10.42.1.10,24

# router por defecto
dhcp-option=3,10.42.1.1

# NTP server
```

```
dhcp-option=option:ntp-server,10.42.1.1

# Search domain
dhcp-option=option:domain-search,co7wt.cu

# dns que usará nuestro cluster
server=1.0.0.1
server=8.8.4.4
```

Aquí aprovechamos y ponemos el anclaje de las MACs, nombres e IP de las PC que queremos, en mi caso agatha es mi laptop, reiniciaremos el servicio luego, veamos como configurar el hostspot.

Activar el hostpot

Usaremos por supuesto Hostapd, lo instalamos con el comando `apt install hpstapd`

Creamos y editamos el fichero de config con el comando `nano /etc/hostapd/hostapd.conf` con el contenido siguiente

```
interface=wlan0
bridge=br0
ssid=PiCluster
country_code=CU
hw_mode=g
channel=12
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=P45w0rd__
wpa_key_mgmt=WPA-PSK
wpa_pairwise=CCMP
wpa_group_rekey=86400
ieee80211n=1
wme_enabled=0
```

Donde evidentemente la red se llamará **«PiCluster»** y el password es **«P45w0rd__»** y usted debe ajustarlo a conveniencia, todavía debemos hacer algunos ajustes, abra el fichero de config por defecto con el comando `nano /etc/default/hostapd`

Y modifique en su contenido la variable DAEMON_CONF para sea igual a esta:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

Ahora habilitemos el servicio con los comandos siguientes

```
systemctl unmask hostapd
systemctl enable hostapd
```

Reinice y verificar que ya el hostspot está activo, conectándolos satisfactoriamente desde una PC/Laptop

Instalar firehol para manejar el enmascaramiento entre las redes del Swarm y las redes de trabajo

Iptables es la óstia, pero yo sigo con firehol que es un wrapper que crea las reglas a partir de un lenguaje más humano, lo instalamos con el comando `apt install firehol`

Vamos a editar el fichero de config con `nano /etc/firehol/firehol.conf` y poner esto:

```
version 6

# Accept all client traffic on any interface
interface any world
    client all accept
    server all accept

router all2all inface any outface any
    masquerade
    route all accept
```

```
client all accept
server all accept
```

Esto permite todas las conex desde/hacia el server en cuaquier interface de red y además activa la ruta enmascarada desde cualquier red hacia cualquier otra red, reiniciando firehol ya debemos tener conectividad

```
firehol restart
```

Por ejemplo si hacemos theter por USB desde el Cell podemos acceder a internet desde la laptop conectada a la WiFi creada por el Raspberry Pi, o desde la WiFi a la red ethernet conectada.

Rompiendo el bloqueo: OpenVPN y configuración del tunel

Si, por culpa de las leyes del bloqueo no podemos acceder a los servicios de docker de manera directa, pero podemos usar un VPN.

Para esto nos hace falta el perfil de la conexión vpn que usemos, casi siempre usamos alguna free y a veces cuando estamos en una universidad y eso debemos usar el proxy, voy a configurarlo todo para que sea los más transparente posible

Partimos de que tengo un fichero *tunnel.ovpn* que tiene las config del tunel y por otro lado las credenciales del mismo.

Si usamos proxy sin credenciales debemos incluir en el fichero *tunnel.ovpn* algo como:

```
http-proxy 10.10.10.34 3128
```

Donde evidentemente **10.10.10.34** es el proxy y 3128 es el puerto, si usas usuario y password pues debes añadir otras configs y quedaría solo así:

```
http-proxy 10.10.10.34 3128 /root/.proxyauth basic
```

Ven que pongo el camino a un fichero `/root/.proxyauth`, este fichero debe contener dos líneas, la primera el username y la segunda el password del proxy

En caso de que el proxy use auth **digest** o **ntlm** solo debes cambiar la palabra **basic** por la que corresponda y listo.

Ahora como hacemos para que no pregunte cada vez por el user y passwd en caso que sea necesario, pues simple, incluir una línea como esta en el profile *tunnel.ovpn*

```
auth-user-pass /root/.vpnauth
```

Donde `/root/.vpnauth` tendrá al igual que en el caso del proxy son dos lineas primeroel username y luego el password

Para iniciar el tunnel es solo llamar al openvpn con la config correspondiente

```
openvpn tunnel.opvn
```

Y ya todas las conexiones que se generen desde CUALQUIERA de las redes del Raspberry Pi serán enviadas vía el tunnel VPN hacia internet

Si, funciona como lo estas pensando… incluso las de la laptop que usas conectada a la WiFi que luego sale por el USB al celular, o vía ethernet al proxy de la universidad… es como si estuvieras directo a internet desde el país por el que sale la VPN que usas, incluso estando detrás de un proxy.

Si quiren lo pueden poner ese comando en un script y llamarlo desde el `/etc/rc.local` para que se mantenga conectado full time ;), ojo en ese caso debes poner un «&» al final de la línea para que se valla a segundo plano.

En al segunda parte abarcaremos todo lo relacionado con montar los workers del nodo con otros SBC y el docker swarm y la configuración de portainerio como gestor.

Cualquier duda ponganla en los comentarios.

Categoría

1. Electronica

Etiquetas

1. cluster
2. docker
3. portainerio
4. Raspberry Pi
5. Swarm
6. VPN

Fecha de creación
junio 2020
Autor
pavelmc