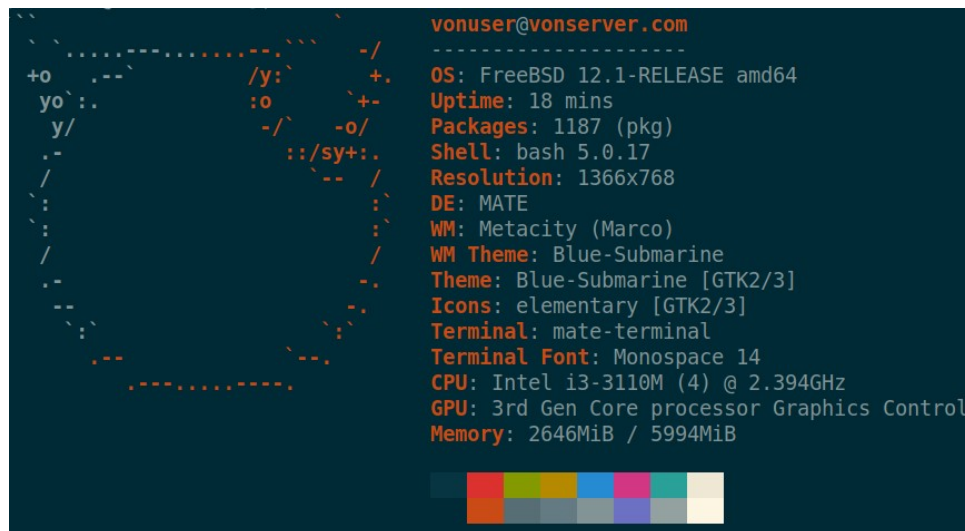


Creación de una máquina virtual con Windows 7 (invitado) utilizando bhyve en FreeBSD 12.1 (anfitrión)

Detalles del Sistema Operativo host:



```
vonuser@vonserver.com
-----
OS: FreeBSD 12.1-RELEASE amd64
Uptime: 18 mins
Packages: 1187 (pkg)
Shell: bash 5.0.17
Resolution: 1366x768
DE: MATE
WM: Metacity (Marco)
WM Theme: Blue-Submarine
Theme: Blue-Submarine [GTK2/3]
Icons: elementary [GTK2/3]
Terminal: mate-terminal
Terminal Font: Monospace 14
CPU: Intel i3-3110M (4) @ 2.394GHz
GPU: 3rd Gen Core processor Graphics Control
Memory: 2646MiB / 5994MiB
```

Ilustración 1: Captura con neofetch

Sistema Utilizado: **Freebsd 12.1-RELEASE FreeBSD 12.1-RELEASE r354233 GENERIC amd64**

Disco Duro: 500 Gb – Ram: 6 Gb

Procesador: Intel i3 – 2 Núcleos virtuales – 2 Núcleos físicos

Nombre de host: **vonserver.com** (obtenido mediante **hostname**). Es importante mencionar que nuestro sistema está basado en el sistema de archivos ZFS. Por otra parte, todas las instrucciones de este tutorial son ejecutadas como usuario *root*.

Hemos descargado previamente una imagen ISO de Windows 7 x 64 bits edición profesional.

Dispositivos de red

re0	Tarjeta de red cableada. Dirección aún no asignada.
wlan0	Tarjeta de red wifi, conectada a un router con conexión a internet (puerta de enlace 192.168.0.1). La configuración es una Ip fija 192.168.0.100/24

Preparación del entorno

Creación de switch virtuales

Configuramos las interfaces virtuales que conectarán nuestra jaula y la máquina virtual contenida en ella. Para ello utilizaremos bhyve. Creamos un dataset en donde se almacenarán las máquinas virtuales del sistema host (anfitrión).

zfs create zroot/vms

Tenemos así el directorio `/zroot/vms`, donde encontraremos la información correspondiente a las máquinas virtuales en nuestro sistema. Sin embargo solo utilizaremos el proceso que brinda `vm` para `networking`, aunque previamente debemos habilitar en `etc/rc.conf`:

`vm_enable="YES"`

`vm_dir=zfs:zroot/vms`

Instalamos los siguientes paquetes:

pkg install -y vm-bhyve bhyve-firmware

Inicializamos los archivos de la máquina virtual mediante:

vm init

Creamos el switch virtual, asignando el nombre `services` y una dirección ip:

vm switch create -a 10.1.1.1/24 services

En cualquier momento podemos monitorear nuestras interfaces mediante **ifconfig**.

Permisos de dispositivos

Este punto, será ampliado en próximas oportunidades. Sin embargo, aquí permitimos mediante reglas en el archivo `/etc/devfs.rules`, que los dispositivos de red y de máquina virtual sean visibles en la jaula. Las siguientes líneas son agregadas; no se modifican las previas.

`[devfs_rules_bhyve_jail=25]`

`add include $devfsrules_jail`

`add path vmm unhide`

```
add path vmm/* unhide
add path tap* unhide
add path zvol/tank/bhyve/* unhide
add path nmdm* unhide
```

Para incorporar nuestras reglas al sistema ejecute **service devfs restart**

Creación de la jaula winJail

Procedemos a crear una jaula o contenedor propio de FreeBSD, a fin de alterar lo menos posible la configuración principal. Por razones prácticas, utilizamos **iocage** para la creación y manejo de jaulas. Instalamos:

```
pkg install py37-iocage
```

Para que se ejecute al inicio de sistema modificamos */etc/rc.conf*:

```
iocage_enable="YES"
```

Activamos la opción *net.link.tap.up_on_open=1*, con **sysctl net.link.tap.up_on_open=1**. En */etc/sysctl.conf*, agregamos simplemente la línea a fin que el sistema siempre active una interface *tap* creada, cada vez que encendamos nuestro equipo. Las interfaces tap son interfaces virtuales, pero operan como nodos de una máquina virtual.

Es recomendable descargar los binarios actualizados para iocage, a fin de siempre disponer de los mismos cuando creemos una jaula:

```
iocage fetch
```

Seleccionamos la versión actual más vigente y procedemos a esperar.

Al finalizar, creamos nuestra jaula virtual:

```
iocage create -r 12.1-RELEASE -n winJail interfaces="vnet0:vm-services" ip4_addr="vnet0|10.1.1.2/24" defaultrouter="10.1.1.1" vnet_default_interface="vm-services" vnet=on allow_vmm=on devfs_ruleset="25"
```

Terminando la instalación tenemos las siguientes recomendaciones para el archivo */etc/sysctl.conf*:

```
net.inet.ip.forwarding=1    # Enable IP forwarding between interfaces
net.link.bridge.pfil_onlyip=0 # Only pass IP packets when pfil is enabled
```

```
net.link.bridge.pfil_bridge=0 # Packet filter on the bridge interface
```

```
net.link.bridge.pfil_member=0 # Packet filter on the member interface
```

Sin embargo, no fueron implementadas para este tutorial.

Creamos una dataset para utilizarlo como un dispositivo zfs donde almacenaremos el disco virtual de windows 7.

```
zfs create -V120G -o volmode=dev zroot/iocage/jails/winJail/root/windata
```

Nuestra configuración final de la jaula es la siguiente (compare mediante **nano /zroot/iocage/jails/winJail/config.json**):

```
{  
  "allow_mount": 1,  
  "allow_mount_devfs": 1,  
  "allow_mount_zfs": 0,  
  "allow_vmm": 1,  
  "cloned_release": "12.1-RELEASE",  
  "defaultrouter": "10.1.1.1",  
  "devfs_ruleset": "25",  
  "enforce_statfs": 1,  
  "host_hostname": "winJail",  
  "host_hostuuid": "winJail",  
  "interfaces": "vnet0:vm-services",  
  "ip4_addr": "vnet0|10.1.1.2/24",  
  "last_started": "2020-07-24 17:07:44",  
  "release": "12.1-RELEASE-p7",  
  "vnet": 1,  
  "vnet0_mac": "xxxxxxxxxxxx xxxxxxxxxxxx",  
  "vnet_default_interface": "vm-services"
```

```
}
```

Traducción de direcciones

La jaula creada no tiene conexión al exterior, es decir, no puede conectarse a internet, motivado a que utiliza un segmento de red que no se encuentra asociado a nuestro dispositivo wifi. Para lograr la conexión es necesario utilizar *Packet Filter*. Editamos el archivo */etc/pf.conf* con el siguiente contenido

```
set skip on lo0  
net_ext=wlan0  
nat on $net_ext from 10.1.1/24 -> wlan0:0
```

Agregamos la configuración correspondiente en */etc/rc.conf*

```
pf_enable="YES"  
pflog_enable="YES"  
pf_rules="/etc/pf.conf"
```

Iniciamos manualmente packet filter con la instrucción:

```
service pf start
```

Dentro de la jaula

Iniciamos la jaula mediante

```
iocage start winJail
```

Comprobando conexión

Comprobamos que nuestra jaula tenga conexión al exterior:

```
ping -c4 8.8.8.8
```

Comprobamos la resolución de nombres mediante:

```
ping -c4 www.google.com
```

```
oot@winJail:~ # ping -c4 www.google.com
PING www.google.com (142.250.64.196): 56 data bytes
4 bytes from 142.250.64.196: icmp_seq=0 ttl=115 time=61.204 ms
4 bytes from 142.250.64.196: icmp_seq=1 ttl=115 time=55.349 ms
4 bytes from 142.250.64.196: icmp_seq=2 ttl=115 time=61.215 ms
4 bytes from 142.250.64.196: icmp_seq=3 ttl=115 time=57.934 ms

-- www.google.com ping statistics --
 4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 55.349/58.926/61.215/2.460 ms
```

Ilustración 2: Comprobación de conexión DNS

Aunque no es obligatorio, configuramos los hosts del sistema al editar el archivo `/etc/hosts`:

```
::1          localhost winjail.com
127.0.0.1    localhost winjail.com
192.168.0.100 vonserver.com (el nombre de nuestro sistema host)
```

Actualización del sistema de paquetes

Para solventar las dependencias requeridas para instalar nuevos paquetes y compilar el árbol de `ports` es necesario actualizar con:

pkg update

Podemos instalar el software requerido para correr nuestra máquina virtual dentro de la jaula:

pkg install -y vm-bhyve nano bhyve-firmware bash

Creamos un directorio que almacene nuestras máquinas virtuales:

mkdir /root/vm

Hacemos los cambios respectivos en `/etc/rc.conf`

```
vm_enable="YES"
```

```
vm_dir="/root/vm"
```

Inicializamos los directorios de `bhyve` con la instrucción **vm init**.

Las instalaciones de windows y el mismo sistema se ejecutan simulando firmware UEFI. El hypervisor byhve trae consigo algunas plantillas que facilitan la configuración de la máquina virtual. Copie el contenido las plantillas de ejemplo al directorio local:

```
cp /usr/local/share/examples/vm-bhyve/* /root/vm/.templates
```

Copiamos el iso de Windows 7 64 bits al subdirectorio .iso de /root/vm, dentro de la jaula. Es decir, que en el host, el directorio destino seria */zroot/iocage/jails/winJail/root/root/vm/.iso/*

Creación del switch de red virtual

Para el manejo de la interfaz de red se crea un switch virtual que se conectará a nuestra interfaz de red virtual principal (vm-services).

```
vm switch create public
```

```
vm switch add public epair0b
```

Escribiendo **vm switch list** verificamos. Si consultamos con ifconfig comprobamos una interfaz llamada vm-public.

```
member: epair0b flags=143<LEARNING,DISCOVER,AUTOEDGE,AUTOPTP>
```

```
groups: bridge vm-switch viid-4c918@
```

Establecemos el shell bash por defecto para el usuario root.

```
chsh -s /usr/local/bin/bash
```

Si no deseamos reiniciar la jaula (**iocage restart winJail**), ejecutamos entonces el entorno de shell bash escribiendo **bash** y presionando enter.

Para asegurar que el switch siempre este disponible para nuestra máquina virtual, hemos creado un script en */root/reiniciar_sw.sh* con el siguiente contenido:

```
#!/bin/bash

#script para reiniciar el switch virtual

echo Eliminando información de switch....

echo >/root/vm/.config/system.conf

echo Creando switch virtual nuevo
```

```
vm switch create public
vm switch add public epair0b
ifconfig vm-public
vm switch list public
```

Para ejecutar el script escribimos

```
sh /root/reiniciar_sw.sh
```

Si la configuración se ejecutó correctamente, obtenemos la siguiente información en pantalla:

NAME	TYPE	IFACE	ADDRESS	PRIVATE	MTU	VLAN	PORTS
public	standard	vm-public	-	no	-	-	epair0b

Si un mensaje indica algún error al agregar la interfaz epair0b al bridge, se recomienda reiniciar la jaula y ejecutar nuevamente el script.

Creando la máquina virtual de windows 7

Mediante la siguiente instrucción, bhyve genera una VM configurada para windows y con 120 Gb de disco duro:

```
vm create -t windows -s 120G win7
```

NAME	DATASTORE	LOADER	CPU	MEMORY	VNC	AUTOSTART	STATE
win7	default	uefi	2	2G	-	No	Stopped

Ilustración 3: Consultando máquinas virtuales creadas con vm list

Para efectuar modificaciones en la configuración podemos modificar el archivo `/root/vm/win7/win7.conf` o a través de la orden **vm config win7**. Cambiamos los siguientes parámetros

cambiamos

```
xhci_mouse="no"
network0_type="virtio-net"
```

agregamos

```
disk0_name="/dev/zvol/zroot/iocage/jails/winJail/root/windata"
disk0_type="virtio-blk"
disk0_dev="custom"
```



```
disk1_name="/root/vm/.iso/virtio-win.iso"
```

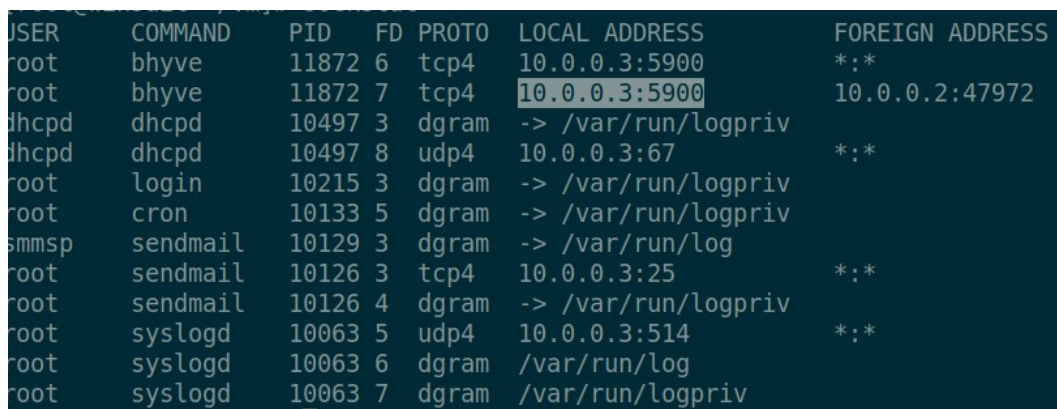
```
disk1_type="ahci-cd"
```

```
disk1_dev="custom"
```

De esta forma hemos agregado un dispositivo de bloque ZFS (ya previamente creado), y una ruta adicional para nuestra imagen iso de drivers virtio descargados de <https://fedorapeople.org/groups/virt/virtio-win/direct-downloads/stable-virtio/virtio-win.iso>.

Procedemos a la instalación de Windows con **vm install win7 win64.iso** Sin embargo necesitaremos buscar un driver de forma automática en la imagen iso (cargada como CD), seleccionar el driver viostore y proceder con la administración de discos. En varias publicaciones relacionadas recomiendan instalar posteriormente, al finalizar la instalación, los drivers Balloon, NetKVM y vioserial (todos en la imagen ISO de virtio drivers). Otra recomendación consiste en ejecutar `qemu-ga-x64.msi installer`. Reinicia el sistema y opera tu versión de Windows 7.

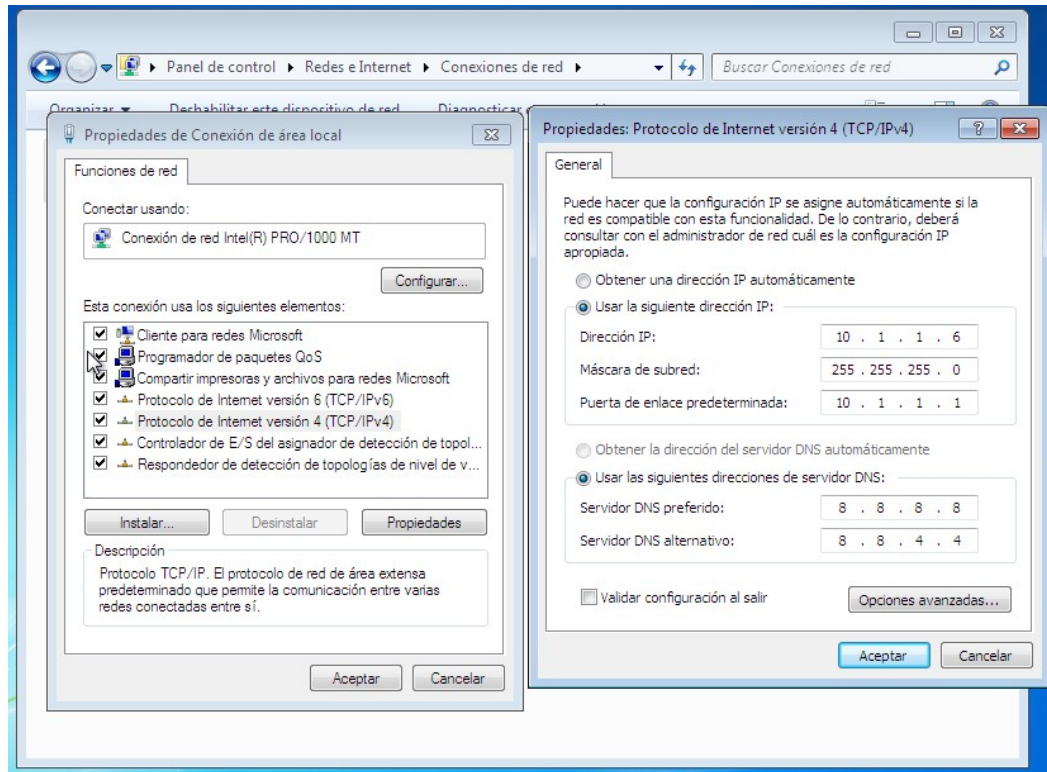
Para monitorear o acceder a la VM utilizamos la aplicación **vncviewer** en la dirección 10.0.0.3:5900, la cual, podemos ejecutar desde nuestro host de FreeBSD.



USER	COMMAND	PID	FD	PROTO	LOCAL ADDRESS	FOREIGN ADDRESS
root	bhyve	11872	6	tcp4	10.0.0.3:5900	*:*
root	bhyve	11872	7	tcp4	10.0.0.3:5900	10.0.0.2:47972
dhcpcd	dhcpcd	10497	3	dgram	-> /var/run/logpriv	
dhcpcd	dhcpcd	10497	8	udp4	10.0.0.3:67	*:*
root	login	10215	3	dgram	-> /var/run/logpriv	
root	cron	10133	5	dgram	-> /var/run/logpriv	
smmsp	sendmail	10129	3	dgram	-> /var/run/log	
root	sendmail	10126	3	tcp4	10.0.0.3:25	*:*
root	sendmail	10126	4	dgram	-> /var/run/logpriv	
root	syslogd	10063	5	udp4	10.0.0.3:514	*:*
root	syslogd	10063	6	dgram	/var/run/log	
root	syslogd	10063	7	dgram	/var/run/logpriv	

Ilustración 4: Consultando puertos abierto con sockstat

Procedemos a la instalación habitual de Windows 7. Ya instalado Windows nuestro sistema no tiene conexión. Abrimos el icono de conexiones de red y configuramos de la siguiente manera:



Espero haber aportado la información necesaria. FreeBSD debe ser para todos.

Referencias

https://github.com/lattera/articles/blob/master/freebsd/2018-10-27_jailed_bhyve/article.md

<https://blog.grem.de/pages/ayvn.html>

<https://dan.langille.org/2015/03/07/getting-started-with-iocage-for-jails-on-freebsd/>

https://wiki.unraid.net/index.php/UnRAID_Manual_6#Creating_Your_Windows_VM

Índice

Detalles del Sistema Operativo host:.....	1
Dispositivos de red.....	1
Preparación del entorno.....	2
Creación de switch virtuales.....	2
Permisos de dispositivos.....	2
Creación de la jaula winJail.....	3
Traducción de direcciones.....	5
Dentro de la jaula.....	5
Comprobando conexión.....	5
Actualización del sistema de paquetes.....	6
Creación del switch de red virtual.....	7
Creando la máquina virtual de windows 7.....	8
Referencias.....	11
Índice.....	12