Guía Rápida de FreeBSD para Usuarios de Linux®

Resumen

Este documento está pensado para que usuarios intermedios y avanzados de Linux® se familiaricen rápidamente con el funcionamiento de FreeBSD.

Tabla de contenidos

1. Introducción

Este documento resalta algunas de las diferencias técnicas entre FreeBSD y Linux® de forma que los usuarios intermedios y avanzados de Linux® se puedan familiarizar con el funcionamiento básico de FreeBSD.

Este documento asume que FreeBSD ya está instalado. Consulta el capítulo Instalar FreeBSD del FreeBSD Handbook para obtener ayuda con el proceso de instalación.

2. Shell por Defecto

Los usuarios de Linux® normalmente se sorprenden al ver que Bash no es el shell por defecto en FreeBSD. De hecho, Bash no se incluye en la instalación por defecto. En su lugar, FreeBSD utiliza tcsh(1) como shell por defecto para root, y el shell compatible con Bourne sh(1) como shell por defecto para los demás usuarios. sh(1) es muy similar a Bash pero con muchas menos funcionalidades. Normalmente los shell scripts escritos para sh(1) se podrán ejecutar en Bash, pero a la inversa no es siempre cierto.

Sin embargo, Bash y otros shells están disponibles para su instalación utilizando los Paquetes y la

Colección de Ports de FreeBSD.

Después de instalar otro shell, utiliza chsh(1) para cambiar el shell por defecto del usuario. Se recomienda que no se cambie el shell por defecto del usuario root ya que los shells que no se incluyen en la distribución base se instalan en /usr/local/bin. En caso de que haya un problema, el sistema de ficheros donde se encuentra /usr/local/bin podría no ser montado. En ese caso, `root` podría no tener acceso a su shell por defecto, evitando que pueda iniciar sesión y arreglar el problema.

3. Paquetes y Ports: Instalar Software en FreeBSD

FreeBSD proporciona dos métodos para instalar aplicaciones: paquetes binarios y ports compilados. Cada método tiene sus propias ventajas:

Paquetes Binarios

- Instalación más rápida comparado con la compilación de aplicaciones de gran tamaño.
- No es necesario saber cómo compilar software.
- No es necesario instalar un compilador.

Ports

- Posibilidad de personalizar las opciones de instalación.
- Se pueden aplicar parches personalizados.

Si la instalación de una aplicación no necesita ninguna personalización, instalar el paquete es suficiente. En cambio, compila el port cuando una aplicación requiera personalización de las opciones por defecto. Si es necesario, se puede compilar un paquete personalizado a partir de ports utilizando `make package`.

Aquí se puede encontrar una lista completa de todos los ports y paquetes disponibles.

3.1. Paquetes

Los paquetes son aplicaciones precompiladas, los equivalentes en FreeBSD de los ficheros .deb de sistemas basados en Debian/Ubuntu y los ficheros .rpm files en los sistemas basados en Red Hat/Fedora. Los paquetes se instalan utilizando `pkg`. Por ejemplo, el siguiente comando instala Apache 2.4:

```
# pkg install apache24
```

Para más información acerca de paquetes consulta la sección 5.4 del FreeBSD Handbook: Usar pkgng para la Gestión de Paquetes Binarios.

3.2. Ports

La Colección de Ports de FreeBSD es un framework de Makefiles y parches específicamente personalizados para instalar aplicaciones con su código fuente en FreeBSD. Al instalar un port, el sistema buscará el código fuente, aplicará los parches necesarios, compilará el código e instalará la aplicación y las dependencias necesarias.

La Colección de Ports, a veces llamada árbol de ports (ports tree), se puede instalar en /usr/ports utilizando Git. Se pueden encontrar instrucciones detalladas para instalar la Colección de Ports en la sección 4.5.1 del FreeBSD Handbook.

Para compilar un port, cambia al directorio del port e inicia el proceso de construcción. El siguiente ejemplo instala Apache 2.4 desde la Colección de Ports:

```
# cd /usr/ports/www/apache24
# make install clean
```

Un beneficio de usar ports para instalar software es la posibilidad de personalizar las opciones de instalación. Este ejemplo especifica que el módulo mod_ldap debería instalarse también:

```
# cd /usr/ports/www/apache24
# make WITH_LDAP="YES" install clean
```

Consulta Usar la Colección de Ports para más información.

4. Inicio del Sistema

Muchas distribuciones Linux® utilizan el sistema de arranque de SysV, mientras que FreeBSD utiliza el init(8) tradicional de BSD. Bajo el init(8) tradicional de BSD, no hay niveles de ejecución (run-levels) y no existe /etc/inittab. En su lugar, el arranque se controla con scripts rc(8). Cuando el sistema arranca, /etc/rc lee /etc/rc.conf y /etc/defaults/rc.conf para determinar qué servicios se tienen que arrancar. Los servicios especificados se arrancan ejecutando el script de inicialización correspondiente situado en /etc/rc.d/ y /usr/local/etc/rc.d/. Estos scripts son parecidos a los que se encuentran en /etc/init.d/ en los sistemas Linux®.

Los scripts que se encuentran en /etc/rc.d/ son para aplicaciones que forman parte del sistema "base", como cron(8), sshd(8), y syslog(3). Los scripts en /usr/local/etc/rc.d/ son para aplicaciones instaladas por el usuario como Apache y Squid.

Puesto que FreeBSD se desarrolla como un sistema operativo completo, las aplicaciones instaladas por el usuario no se consideran parte del sistema "base". Las aplicaciones instaladas por el usuario normalmente se instalan mediante Paquetes o Ports. Para mantenerlos separados del sistema base, estas aplicaciones se instalan en /usr/local/. Por lo tanto, los binarios instalados por el usuario se encuentran en /usr/local/bin/, los ficheros de configuración en /usr/local/etc/, y así sucesivamente.

Los servicios se habilitan añadiendo una entrada para el servicio en /etc/rc.conf. Los

predeterminados del sistema se encuentran en /etc/defaults/rc.conf a los que se anteponen los servicios en /etc/rc.conf. Consulta rc.conf(5) para más información acerca de las entradas disponibles. Cuando instales aplicaciones adicionales, revisa los mensajes de instalación de la aplicación para determinar cómo activar cualquier servicio que tenga asociado.

Las siguientes entradas en /etc/rc.conf activan sshd(8), activan Apache 2.4, y especifican que Apache debería arrancar con SSL.

```
# activa SSHD
sshd_enable="YES"
# activa Apache con SSL
apache24_enable="YES"
apache24_flags="-DSSL"
```

Una vez que un servicio ha sido activado en /etc/rc.conf, puede iniciarse sin reiniciar el sistema:

```
# service sshd start
# service apache24 start
```

Si un servicio no se ha habilitado, se puede arrancar desde la línea de comando utilizando `onestart`:

```
# service sshd onestart
```

5. Configuración de la Red

En lugar de un identificador genérico *ethX* que Linux® utiliza para identificar un interfaz de red, FreeBSD utiliza el nombre del controlador seguido de un número. Lo siguiente salida del comando ifconfig(8) muestra dos interfaces de red Intel® Pro 1000 (em0 y em1):

```
% ifconfig
em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=b<RXCSUM,TXCSUM,VLAN_MTU>
        inet 10.10.10.100 netmask 0xffffff00 broadcast 10.10.10.255
        ether 00:50:56:a7:70:b2
        media: Ethernet autoselect (1000baseTX <full-duplex>)
        status: active
em1: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=b<RXCSUM,TXCSUM,VLAN_MTU>
        inet 192.168.10.222 netmask 0xffffff00 broadcast 192.168.10.255
        ether 00:50:56:a7:03:2b
        media: Ethernet autoselect (1000baseTX <full-duplex>)
        status: active
```

Se puede asignar una dirección IP a un interfaz utilizando ifconfig(8). Para que permanezca de forma persistente entre arranques, la configuración IP se debe incluir en /etc/rc.conf. Las siguientes

entradas en /etc/rc.conf especifican el nombre del host, dirección IP, y la pasarela por defecto:

```
hostname="server1.example.com"
ifconfig_em0="inet 10.10.10.100 netmask 255.255.255.0"
defaultrouter="10.10.10.1"
```

Utiliza las siguientes entradas si en su lugar quieres configurar un interfaz con DHCP:

```
hostname="server1.example.com"
ifconfig_em0="DHCP"
```

6. Firewall

FreeBSD no utiliza como firewall el IPTABLES de Linux®. En su lugar, FreeBSD ofrece tres posibles firewalls a nivel de kernel:

- PF
- IPFILTER
- IPFW

PF es desarrollado por el proyecto OpenBSD y ha sido portado a FreeBSD. PF se creó como reemplazo de IPFILTER y su sintaxis es muy similar a la que tenía este último. PF se puede usar junto con altq(4) para proporcionar características QoS.

Este ejemplo de entrada de PF permite conexiones entrantes SSH:

```
pass in on $ext_if inet proto tcp from any to ($ext_if) port 22
```

IPFILTER es el firewall desarrollado por Darren Reed. No es específico de FreeBSD y se ha portado a varios sistemas operativos, incluidos NetBSD, OpenBSD, SunOS, HP/UX y Solaris.

La sintaxis de IPFILTER utilizada para permitir conexiones entrantes de SSH es:

```
pass in on $ext_if proto tcp from any to any port = 22
```

IPFW es el firewall desarrollado y mantenido por FreeBSD. Se puede utilizar junto con dummynet(4) para proporcionar capacidades de perfilado de tráfico y simular distintos tipos de conexiones de red.

La sintaxis de IPFW para permitir conexiones entrantes de SSH sería:

```
ipfw add allow tcp from any to me 22 in via $ext_if
```

7. Actualizando FreeBSD

Hay dos métodos para actualizar un sistema FreeBSD: a partir del código fuente o mediante la actualización de los binarios.

Actualizar desde código fuente es el método más complejo pero el que ofrece mayor flexibilidad. El proceso implica la sincronización de una copia local del código fuente de FreeBSD con los servidores Subversion de FreeBSD. Una vez actualizado el código fuente, se pueden compilar nuevas versiones de las utilidades y el kernel.

Las actualizaciones binarias son similares a utilizar `yum` o `apt-get` para actualizar un sistema Linux®. En FreeBSD, se puede utilizar freebsd-update(8) para obtener e instalar actualizaciones binarias. Estas actualizaciones se pueden programar utilizando cron(8).

Cuando se use cron(8) para planificar actualizaciones, utiliza `freebsd-update cron` en crontab(1) para reducir la posibilidad de un gran número de máquinas obteniendo las actualizaciones todas al mismo tiempo:

```
0 3 * * * root /usr/sbin/freebsd-update cron
```

Para más información sobre las actualizaciones binarias, consulta el capítulo sobre actualizaciones en el FreeBSD Handbook.

8. procfs: Desaparecido Pero No Olvidado

En algunas distribuciones Linux®, uno puede mirar en /proc/sys/net/ipv4/ip_forward para determinar si IP forwarding está activado. En FreeBSD, se usa sysctl(8) para ver esta y otras configuraciones del sistema.

Por ejemplo, utiliza lo siguiente para determinar si IP forwarding está activado en un sistema FreeBSD:

```
% sysctl net.inet.ip.forwarding
net.inet.ip.forwarding: 0
```

Usa `-a` para listar toda la configuración del sistema:

```
% sysctl -a | more
```

Si una aplicación necesita procfs, añade la siguiente línea a /etc/fstab:

```
proc              /proc          procfs  rw,noauto      0       0
```

Incluir `noauto` impedirá que /proc se monte de forma automática al arrancar.

Para montar el sistema de archivos sin reiniciar:

```
# mount /proc
```

9. Comandos Comunes

Algunos equivalentes de los comandos comunes son los siguientes:

Comando Linux® (Red Hat/Debian)	Equivalente FreeBSD	Propósito
`yum install package` / `apt-get install package`	`pkg install package`	Instalar un paquete de un repositorio remoto
`rpm -ivh package` / `dpkg -i package`	`pkg add package`	Instalar un paquete local
`rpm -qa` / `dpkg -l`	`pkg info`	Listar paquetes instalados
`lspci`	`pciconf`	Listar dispositivos PCI
`lsmod`	`kldstat`	Listar módulos del kernel cargados
`modprobe`	`kldload` / `kldunload`	Cargar/Descargar módulos del kernel
`strace`	`truss`	Trazar llamadas al sistema

10. Conclusión

Este documento ha proporcionado un resumen de FreeBSD. Consulta el FreeBSD Handbook para una cobertura más en profundidad sobre estos temas así como de muchos de los temas no cubiertos en este documento.