FreeBSD y los Dispositivos de Estado Sólido (SSD)

Resumen

Este artículo trata sobre el uso de discos de estado sólido en FreeBSD para crear sistemas embebidos.

Los sistemas embebidos tienen la ventaja de una mayor estabilidad por la falta de partes móviles (discos duros). Sin embargo, se debe tener en cuenta que generalmente el espacio disponible para el sistema y la durabilidad del medio de almacenamiento son menores.

Los temas específicos que se abordarán incluyen los tipos y atributos de los dispositivos de estado sólido adecuados para su uso como disco en FreeBSD, las opciones del kernel que son interesantes para dicho entorno, los mecanismos de rc.initdiskless que automatizan el inicio de dichos sistemas, la necesidad de sistemas de archivos de solo lectura y hacer sistemas de archivos desde cero. El artículo concluirá con algunas estrategias generales para entornos pequeños y de solo lectura de FreeBSD.

Tabla de contenidos

1. Dispositivos de Disco de Estado Sólido

El alcance de este artículo se limitará a dispositivos de estado sólido basados en memoria flash. La memoria flash es una memoria de estado sólido (sin partes móviles) que no es volátil (la memoria mantiene los datos incluso después de que se hayan desconectado todas las fuentes de alimentación). La memoria flash puede soportar un enorme impacto físico y es bastante rápida (las soluciones de memoria flash que se tratan en este artículo son un poco más lentas que un disco duro EIDE en operaciones de escritura y mucho más rápidos en operaciones de lectura). Un aspecto muy importante de la memoria flash, cuyas repercusiones se tratarán más adelante, es que cada sector tiene una capacidad de reescritura limitada. Solo puede escribir, borrar y volver a escribir en un sector de la memoria flash varias veces antes de que quede permanentemente inutilizable. Aunque muchos productos de memoria flash mapean automáticamente los bloques defectuosos y algunos incluso distribuyen las operaciones de escritura de manera uniforme en toda la unidad, la verdad es que hay un límite en la cantidad de escrituras que se pueden hacer al dispositivo. Las unidades más competitivas tienen entre 1.000.000 y 10.000.000 millones de escrituras por sector en

sus especificaciones. Esta cifra varía debido a la temperatura del ambiente.

Específicamente, discutiremos las unidades flash compactas compatibles con ATA, las cuales son bastante populares como medios de almacenamiento para cámaras digitales. Es de particular interés el hecho de que se conecten directamente al bus IDE y sean compatibles con el conjunto de comandos ATA. Por lo tanto, con un adaptador muy simple y de bajo coste, estos dispositivos se pueden conectar directamente al bus IDE en un ordenador. Una vez implementado de esta forma, los sistemas operativos como FreeBSD ven el dispositivo como un disco duro normal (aunque sea pequeño).

Existen otras soluciones de disco de estado sólido, pero su coste, opacidad y su relativa dificultad de uso los colocan más allá del alcance de este artículo.

2. Opciones del Kernel

Algunas opciones del kernel son de especial interés para aquellos que crean un sistema FreeBSD embebido.

Todos los sistemas FreeBSD embebidos que usen memoria flash como disco del sistema estarán interesados en los discos y sistemas de fichero en memoria. Como resultado del limitado número de escrituras que se pueden hacer a la memoria flash, el disco y los sistemas de fichero del disco se montarán seguramente como solo lectura. En este entornos, sistemas de ficheros como /tmp y /var se montan como sistemas de fichero en memoria para permitir que el sistema cree logs y actualice contadores y ficheros temporales. Los sistemas de ficheros en memoria son un componente crítico para una implementación de estado sólido exitosa en FreeBSD.

Deberías asegurarte de que las siguientes líneas están en el fichero de configuración de tu kernel:

```
options         MFS             # Memory Filesystem
options         MD_ROOT         # md device usable as a potential root device
pseudo-device   md              # memory disk
```

3. El Subsistema `rc` y los Sistemas de ficheros de Solo Lectura

La inicialización posterior al arranque de un sistema FreeBSD embebido es controlada por /etc/rc.initdiskless.

/etc/rc.d/var monta /var como un sistema de ficheros en memoria, crea una lista configurable de directorios en /var con el comando mkdir(1), y cambia los modos de algunos de esos directorios. En la ejecución de /etc/rc.d/var, otra variable de rc.conf entra en juego - `varsize`. Una partición /var es creada por /etc/rc.d/var basándose en el valor de esta variable en rc.conf:

```
varsize=8192
```

Recuerda que por defecto este valor está en sectores.

El hecho de que /var sea un sistema de archivos de lectura y escritura es una distinción importante, ya que la partición / (y cualquier otra partición que puedas tener en tu medio flash) se debe montar como solo lectura. Recuerda que en Dispositivos de Disco de Estado Sólido detallamos las limitaciones de la memoria flash, específicamente, la capacidad de escritura limitada. La importancia de no montar sistemas de archivos en medios flash de lectura-escritura, y la importancia de no usar swap, no es exagerada. Un archivo swap en un sistema concurrido puede deteriorar un medio flash en menos de un año. Un logging intenso o la creación y destrucción de archivos temporales puede hacer lo mismo. Por lo tanto, además de quitar la entrada `swap` de tu /etc/fstab, también deberías cambiar el campo Options para cada sistema de archivos a `ro` de la siguiente forma:

```
# Device                Mountpoint      FStype  Options         Dump    Pass#
/dev/ad0s1a             /               ufs     ro              1       1
```

Algunas aplicaciones en el sistema comenzarán a fallar inmediatamente como resultado de este cambio. Por ejemplo, cron no se ejecutará correctamente al faltar las cron tabs en /var creadas por /etc/rc.d/var, además, syslog y dhcp encontrarán problemas como resultado de montar el sistema de archivos como solo lectura y la falta de elementos en /var que ha creado /etc/rc.d/var. Sin embargo, esto son solo problemas temporales y se tratan, junto con las soluciones para la ejecución de otros programas de uso común en Estrategias para Entornos Pequeños y de Solo Lectura.

Una cosa importante a recordar es que un sistema de archivos que fue montado como solo lectura con /etc/fstab puede ser montado como lectura-escrita en cualquier momento ejecutando el comando:

```
# /sbin/mount -uw partition
```

y se puede cambiar de nuevo a solo lectura con el comando:

```
# /sbin/mount -ur partition
```

4. Construyendo un Sistema de Archivos Desde Cero

Como las tarjetas flash compactas compatibles con ATA son vistas por FreeBSD como discos duros IDE normales, en teoría podrías instalar FreeBSD desde una red usando los disquetes kern y mfsroot o desde un CD.

Sin embargo, incluso una instalación pequeña de FreeBSD usando el procedimiento normal de instalación puede producir un sistema de tamaño superior a los 200 megabytes. La mayoría de la gente usará memorias flash menores (128 megabytes se considera bastante grande - 32 o incluso 16 megabytes es bastante común), de forma que una instalación utilizando mecanismos normales no

es posible - simplemente no hay suficiente espacio en disco incluso para la más pequeña de las instalaciones convencionales.

La forma más fácil de superar esta limitación de espacio es instalar FreeBSD utilizando medios convencionales en un disco duro normal. Una vez finalizada la instalación, reduce el sistema operativo a un tamaño que se ajuste a tu medio flash, y comprime el sistema de archivos completo en un fichero tar. Los siguientes pasos te guiarán en el proceso de preparación de una memoria flash para tu sistema de archivos comprimido en un fichero tar. Recuerda que no estamos ejecutando una instalación normal, luego las operaciones como particionado, etiquetado, creación del sistema de archivos, etc. deben ejecutarse manualmente. Además de los disquetes kern y mfsroot, también necesitarás usar el disquete fixit.

1. Particionando Tu Dispositivo Flash

   Después de arrancar con los disquetes kern y mfsroot, selecciona `custom` en el menú de instalación. En el menú de instalación personalizado, selecciona `partition`. En el menú de particiones, debe borrar todas las particiones existentes mediante la tecla `d`. Después de eliminar todas las particiones existentes, crea una partición utilizando la tecla `c` y acepta el valor predeterminado para el tamaño de la partición. Cuando se te pregunte el tipo de partición, asegúrate de que el valor esté establecido en `165`. Ahora escribe la tabla de particiones en el disco presionando `w` (es una opción oculta en esta pantalla). Si estás utilizando una tarjeta compact flash compatible con ATA, debes elegir el FreeBSD Boot Manager. Ahora presiona `q` para salir del menú de partición. Verás de nuevo el menú del gestor de arranque - repite la opción hecha anteriormente.

2. Creación de Sistemas de Archivos en Tu Dispositivo de Memoria Flash

   Sal del menú de instalación personalizada, y desde el menú principal de instalación escoge la opción `fixit`. Después de entrar en el entorno fixit, introduce el siguiente comando:

   ```
   # disklabel -e /dev/ad0c
   ```

   En este punto, habrás accedido al editor vi guiado por el comando disklabel. A continuación, debes agregar una línea `a:` al final del archivo. La línea `a:` debería ser similar a la siguiente:

   ```
   a:      123456  0       4.2BSD  0       0
   ```

   Donde *123456* es un número que es exactamente el mismo número en la entrada `c:` existente para el tamaño. Básicamente estás duplicando la línea `c:` existente como una línea `a:`, asegurándote de que el fstype es `4.2BSD`. Salva el fichero y sal.

   ```
   # disklabel -B -r /dev/ad0c
   # newfs /dev/ad0a
   ```

3. Colocando Tu Sistema de Archivos en el Medio Flash

Monta el medio flash recién preparado:

```
# mount /dev/ad0a /flash
```

Coloca esta máquina en la red para poder transferir nuestro archivo tar y extraerlo en nuestro sistema de archivos del medio flash. Un ejemplo de cómo hacerlo es:

```
# ifconfig xl0 192.168.0.10 netmask 255.255.255.0
# route add default 192.168.0.1
```

Ahora que la máquina está en la red, transfiere tu archivo tar. Es posible que te enfrentes a un pequeño dilema en este punto - si tu memoria flash tiene por ejemplo 128 megabytes, y tu archivo tar tiene más de 64 megabytes, no podrás tener el archivo tar en el medio de flash al mismo tiempo que realizas la descompresión - te quedarás sin espacio. Una solución a este problema, si estás utilizando FTP, es descomprimir el archivo mientras se transfiere por FTP. Si realizas la transferencia de esta forma, nunca tendrás el archivo tar y los contenidos en el disco al mismo tiempo:

```
ftp> get tarfile.tar "| tar xvf -"
```

Si tu archivo tar está comprimido con gzip, puedes hacerlo de esta forma:

```
ftp> get tarfile.tar "| zcat | tar xvf -"
```

Una vez que el contenido de tu sistema de archivos comprimido por tar está en el sistema de archivos de la memoria flash, puedes desmontar la memoria flash y reiniciar:

```
# cd /
# umount /flash
# exit
```

Suponiendo que configuraste correctamente tu sistema de archivos cuando lo construiste en tu disco duro normal, (con tus sistemas de archivos montados en modo solo lectura, y con las opciones necesarias compiladas en el kernel) ahora se deberías iniciar con éxito tu sistema embebido FreeBSD.

5. Estrategias para Entornos Pequeños y de Solo Lectura

En El Subsistema `rc` y los Sistemas de ficheros de Solo Lectura, se indicó que el sistema de archivos /var construido por /etc/rc.d/var y la presencia de un sistema de archivos raíz montado en modo

solo lectura causa problemas con muchos paquetes de software utilizados en FreeBSD. En este artículo, se proporcionarán sugerencias para ejecutar con éxito cron, syslog, la instalación de ports y el servidor web Apache.

5.1. Cron

Durante el arranque, /etc/rc.d/var puebla /var usando la lista de /etc/mtree/BSD.var.dist, de forma que se crean cron, cron/tabs, at, y otros pocos directorios estándar.

Sin embargo, esto no resuelve el problema de mantener las cron tabs entre los reinicios. Cuando el sistema se reinicie, el sistema de archivos /var cargado en memoria desaparecerá y todas las cron tabs que tenga también desaparecerán. Por lo tanto, una solución sería crear las cron tabs para los usuarios que las necesiten; monta tu sistema de archivos raíz / como lectura-escritura y copia las cron tabs a un lugar seguro, como /etc/tabs, a continuación, añade una entrada al final de /etc/rc.initdiskless que copie estas crontabs a /var/cron/tabs después de que el directorio se cree durante el inicio del sistema. Es posible que también debas añadir una entrada que cambie los modos y permisos en los directorios creados y en los archivos copiados con /etc/rc.initdiskless.

5.2. Syslog

syslog.conf especifica las ubicaciones de ciertos ficheros de log que hay en /var/log. Estos archivos no son creados por /etc/rc.d/var durante la inicialización del sistema. Por lo tanto, en algún lugar de /etc/rc.d/var, justo después de la sección que crea los directorios en /var, tendrás que añadir algo como esto:

```
# touch /var/log/security /var/log/maillog /var/log/cron /var/log/messages
# chmod 0644 /var/log/*
```

5.3. Instalación de ports

Antes de analizar los cambios necesarios para utilizar con éxito el árbol de ports, es necesario recordar que su sistema de archivos en el medio flash es de solo lectura. Dado que es de solo lectura, necesitarás montarlo temporalmente en modo lectura-escritura utilizando la sintaxis que se muestra en El Subsistema `rc` y los Sistemas de ficheros de Solo Lectura. Siempre debes volver a montar estos sistemas de archivos en modo solo lectura cuando hayas terminado cualquier mantenimiento - las escrituras innecesarias en el medio flash podrían acortar considerablemente su vida útil.

Para que sea posible entrar en un directorio de ports y ejecutar con éxito `make install`, debemos crear un directorio de paquetes en un sistema de ficheros que no esté en memoria que seguirá la pista de nuestros paquetes entre reinicios. Como de todos modos es necesario montar tus sistemas de ficheros como lectura-escritura para la instalación de paquetes, parece razonable asumir que se puede usar un área del medio flash para que se escriba información de los paquetes.

Primero, crea el directorio para la base de datos de los paquetes. Normalmente se encuentra en /var/db/pkg, pero no podemos colocarlo allí ya que desaparecerá cada vez que se inicie el sistema.

```
# mkdir /etc/pkg
```

Ahora, añade una línea a /etc/rc.d/var que enlace el directorio /etc/pkg a /var/db/pkg. Un ejemplo:

```
# ln -s /etc/pkg /var/db/pkg
```

Ahora, cada vez que montes tus sistemas de ficheros como lectura-escritura e instales un paquete, `make install` funcionará, se escribirá la información del paquete con éxito en /etc/pkg (porque el sistema de ficheros se montará, en su momento, como sólo lectura) que estará siempre disponible para el sistema operativo como /var/db/pkg.

5.4. Servidor Web Apache

Los pasos de esta sección solo son necesarios si Apache está configurado para escribir su pid o registro log fuera de /var. Por defecto, Apache guarda su archivo pid en /var/run/httpd.pid y sus registros de log en /var/log.

Ahora se asume que Apache mantiene sus ficheros de log en un directorio apache_log_dir fuera de /var. Cuando este directorio vive en un sistema de ficheros de sólo lectura, Apache no será capaz de guardar ningún fichero de log, y podría tener problemas de funcionamiento. Si es así, es necesario añadir un nuevo directorio a la lista de directorios en /etc/rc.d/var para crear en /var, y para enlazar apache_log_dir a /var/log/apache. También es necesario establecer permisos y el propietario de este nuevo directorio.

Primero, añade el directorio `log/apache` a la lista de directorios para ser creados en /etc/rc.d/var.

En segundo lugar, agrega estos comandos a /etc/rc.d/var después de la sección de creación del directorio:

```
# chmod 0774 /var/log/apache
# chown nobody:nobody /var/log/apache
```

Por último, elimina el directorio apache_log_dir existente, y reemplázalo con un enlace:

```
# rm -rf apache_log_dir
# ln -s /var/log/apache apache_log_dir
```