COMP3301 Assignment 3

OpenBSD VHD Kernel Driver - Filing the System

Due: 3pm Monday in Week 13 (21st of October)
Submission: Git
Code submission is marked in your prac session in week 13
Last Updated: October 12, 2024

1 Academic Integrity

All assessments are individual. You should feel free to discuss aspects of C programming and assessment specifications with fellow students and discuss the related APIs in general terms. You should not actively help (or seek help from) other students with the actual coding of your assessment. It is cheating to look at another student's code, and it is cheating to allow your code to be seen or shared in printed or electronic form. You should note that all submitted code will be subject to automated checks for plagiarism and collusion. If we detect plagiarism or collusion (outside of t[illegible]t proceedings will be initiated against you. If [illegible]taff member. Do not be tempted to copy anc[illegible]stand the statements on student misconduct i[illegible]ps://eecs.uq.edu.au/current-students/[illegible]conduct

1.1 Use of AI T[illegible]

All assessment tasks eva[illegible]thout the aid of generative Artificial Intellig[illegible] are advised that the use of AI technologies t[illegible]**tly prohibited** and may constitute student [illegible]

2 Backgroun[illegible]

This assignment extend[illegible]VHD disk images as block devices. This is si[illegible]`vnd(4)` driver, which supports using files containing a disk image as a block device. The purpose of this assignment is to exercise concepts of low-level disk operations and caching in an operating system kernel environment.

From a high-level point of view, a physical disk device presents a sequence of bytes that can be written to or read from, with the ability to quickly seek an arbitrary position and read and write at that point. Note that this is a simplification that ignores that disks address and provide access to blocks of bytes, not individual bytes.

A file on most operating systems offers similar features, i.e., a sequence of bytes that can be accessed by address. Because of these similarities, it is possible for an operating system to provide a common set of operations on both files and disks (e.g., open, close, read, write, seek, etc.) and allow them to be used interchangeably. For example, you could use `tar` to write an archive to a file in a filesystem, or directly to a disk device. `dd`, `cp`, `cat`, etc can read the bytes from a raw disk into a file or visa versa. However, operating systems generally provide extra functionality on top of disk devices such as the ability to partition disks and mount filesystems from them.

2.1 vnd(4)

The `vnd(4)` driver in OpenBSD provides a ”disk-like interface to a file”. This means the OpenBSD kernel can open a file and present it as a block device to the rest of the system, which in turn allows for the creation and use of filesystems on these disk images.

The `vnd(4)` driver currently only supports using raw disk images as backing files. There’s a one-to-one mapping of data offsets for data in the end disk device and the byte offset of that data in the underlying file. This makes the implementation very simple, with the downside that the backing file has to be the same size as the disk `vnd` is presenting. If you have a 32G disk image, the file will be 32G regardless of how much data is actually stored inside a filesystem mounted on top of it. Similar functionality exists in the `loop` driver in Linux, and the `lofi` driver in Solaris and Illu

2.2 Virtual Har

Virtual Hard Disk (VH nectix’s Virtual PC
hypervisor, which was l osoft Virtual PC. A
defining feature of VHD ze of the file backing
a disk image on deman e backing file has to
pre-allocate space for th also have fixed-size
images, works in a very

VHD also supports usi writing changes to a
separate file. When use g of virtual machines
where only changes ma n repeated copies of
disks. Due to the popul mon for disk images
to be distributed as VH

The VHD format used l
https://stluc.manta.uqcloud.net/comp3301/public/2024/comp3301-vhd-spec.pdf

The same file can be found on Blackboard. This specification differs from the official Microsoft specification in that it includes annotations for ease of understanding and contains corrections to errors in the official specification. The official VHD format is documented by Microsoft in `Virtual Hard Disk Format Spec_10_18_06.doc`.

3 Instructions

To complete the assignment, you will need to do the following:

1. Download the base code patch

```
cd ~
ftp https://stluc.manta.uqcloud.net/comp3301/public/2024/comp3301-2024-
    a3.patch
```

2. Create the a3 branch

```
cd /usr/src
git checkout -b a3 openbsd-7.5
```

3. Apply the base code patch

```
git am < ~/comp3301-2024-a3.patch
```

4. Install the includes

```
cd /usr/src/include
doas make includes
```

5. Build the kernel

```
cd /usr/src/sys
make obj
make config
make -j4
doas make insta
```

6. Reboot

```
doas reboot
```

7. Build and install v

```
cd /usr/src/usr
make obj
make
doas make insta
```

8. Create vhd(4) dev

```
doas cp /usr/src/etc/etc.amd64/MAKEDEV /dev
cd /dev
doas chmod 755 MAKEDEV
doas ./MAKEDEV vhd0
```

4 Specifications

You will be extending the OpenBSD kernel to add support for using VHD files as a backend for a vhd(4) virtual block device. vhd(4) is roughly based on vnd(4). Boilerplate code for the device entry points and command line utility will be provided, but you will be implementing the handling of the file and the VHD file format within the provided kernel driver.

Only a subset of the VHD functionality listed in Microsoft's specification of the file format is required. The following functionality is required:

- Read support
- Write support
- Fixed-size images
- Dynamic images

Differencing images do not need to be supported. In addition to supporting the VHD file format, the kernel should implement the following:

- Deny attaching VHD files which are invalid, corrupted or contain features not supported by your kernel driver.
- Deny detaching VHD files when the disk is open unless the force flag is passed to the `VHDIOCDETACH` `ioctl`.
- Return the name of the VHD file the device was attached to for the `VHDIOCFNAME` `ioctl`.
- Populate a `struct stat` for the currently attached VHD file for the `VHDIOCSTAT` `ioctl`.

4.1 VHD Disk I/O

Reads and writes again [illegible] `_rdwr()` against the VHD backing file. Writ [illegible] ng VHD disk image. **The read and write f [illegible] the marks for this assignment.**

Caching of VHD structu [illegible] allocation table) and data blocks should be i [illegible] erformance. For example, if you read a sec [illegible] other sector from the same block, you should [illegible] Same goes for mixed reads and writes, for ex [illegible] e on the same block should result in one rea [illegible] number of data block cached, the cache replac [illegible] more blocks) and the implementation details [illegible] some form of caching is implemented.

4.2 `ioctl` interfa [illegible]

The following `ioctl`s should only work on the raw partition of the character device associated with each `vhd` disk. Except for `VHDIOCATTACH`, they should only work when a `vhd` disk is attached to a backing file.

`VHDIOCATTACH`

Specify the VHD file to attach as a block device, and parameters for using the disk in the kernel. The `vhd_attach` struct contains the following fields:

- `vhd_file` - The name of the VHD file to attach a `vhd` disk to.
- `vhd_readonly` - The `vhd` disk can be written to when set to 0, and if the VHD file is read-only, the `vhd` disk should fail to attach. The `vhd` disk should be read-only when set to a non-zero value, and the VHD file should be opened read-only.

VHDIOCDETACH

This ioctl requests the block device be detached, and the backing file closed. If the disk is in use, the request should fail with EBUSY unless the unsigned int ioctl argument is set to a non-zero value. A non zero value requests the forced removal of the block device and close of the backing VHD file.

VHDIOCFNAME

This ioctl requests the name of the VHD file used for the currently attached block device. The name should be the same as what was passed as the filename in the VHDIOCATTACH ioctl.

VHDIOCSTAT

This ioctl is equivalent to an fstat(2) system call against the backing file.

5 Provided Tools/Files

5.1 vhdctl

The patch includes sour ned above to control vhd devices. It is simila e patch is applied in src/usr.sbin/vhdctl.

5.2 vhdtool

vhdtool allows you to c aw and VHD. It also allows you to perform b o dump their footers and dynamic disk heade It may be installed by running the following

```
cd ~
ftp https://stluc.m                                   tool.tar
doas tar -xpf vhdto
```

5.3 rawtest.img

This is a raw disk of size 10 MiB which contains a filesystem with the following files:

```
-rw-r--r--   1 root  wheel    10 Oct  4 21:37 comp3301.txt
drwxr-xr-x   2 root  wheel   512 Oct  4 21:39 folder/
-rwxr-xr-x   1 root  wheel  6496 Oct  4 21:40 hello*
-rw-r--r--   1 root  wheel    94 Oct  4 21:40 hello.c
-rw-r--r--   1 root  wheel    13 Oct  4 21:36 hello.txt
-rw-r--r--   1 root  wheel  2739 Oct  4 21:38 test.txt
```

Download:
https://stluc.manta.uqcloud.net/comp3301/public/2024/rawtest.img.