Overview:

Our implementation of XCFS was strongly derived from the ecryptfs file system, with some portions coming from the wrapfs section. The only bug we weren't able to fix was that the writepage() function doesn't ever get called, so the file system doesn't write explicitly mmapped pages from the page cache back to disk. We were able to verify this behavior using a combination of mmap and msync in a custom c program.

Files:

The files included in this project are as follows:

- dentry.c – code related to the dentry structs and their related functions. This code was copied from wrapfs with minimal modifications.
- file.c – code related to file structs and their related functions. This code was copied from wrapfs with extensive changes to read() and write() to support buffered IO encryption/decryption.
- inode.c – code related to inodes and their related functions. This code was copied from wrapfs with minimal modifications.
- lookup.c – code related to looking through directories. This code was copied from wrapfs with minimal modifications.
- main.c – initialization functions
- mmap.c – code related to memory mapping. The implementation of readpage() draws heavily from ecryptfs's implementation (with a few modifications due to differences in what metadata our implementation keeps in the inode struct), while the implementation of writepage() differs significantly (for the same reason we had to modify the readpage() implementation) – instead of calling kernel_write(), we had to pass the page down to the lower filesystem's implementation of writepage().
- super.c – code related to superblocks. This was copied from wrapfs with minimal modifications.
- xcfs.h – header file with function declarations, global variables, and #defines

Testing:

First, insert the module and mount the filesystem.

```
root@KITT-QEMU:~# insmod xcfs.ko
root@KITT-QEMU:~# mount -t xcfs test test-xcfs/
root@KITT-QEMU:~# ls
123      a.out      test    testmmap.c  test-xcfs
123xcfs  hello.txt  test.c  test.sh     xcfs.ko
root@KITT-QEMU:~# ls test
normal-file
root@KITT-QEMU:~# ls test-xcfs/
normal-file
root@KITT-QEMU:~#
```

Now copy a test.c file over to the filesystem:

```
root@KITT-QEMU:~# cp test.c test-xcfs/
root@KITT-QEMU:~# cat test-xcfs/test.c
#include <stdio.h>

int main(int argc, char **argv) {
        printf("Hello world\n");
        return 0;
}
root@KITT-QEMU:~# cat test/test.c
$jodmvef!=tuejp/i?

                  jou!nbjo)jou!bshd-!dibs!++bshw*!|

qsjoug)#Ifmmp!xpsme]o#*<

sfuvso!1<
         ~
          root@KITT-QEMU:~#
```

Compile the program and run it:

```
root@KITT-QEMU:~/test-xcfs# gcc test.c
root@KITT-QEMU:~/test-xcfs# ./a.out
Hello world
root@KITT-QEMU:~/test-xcfs#
```