NOTE: Any changes to the code that I have made in cod has comments to help locate them easily. Simply serch for “ADDED by Avinash Srinivasan -->”. I have also added several print statements to just follow the flow of code.

In the retrieve\_file() function in file\_hiding\_helper.C, I have added the below block to replace what was there originally ()which is not commented out.

sprintf(cmd, "sudo bmap --mode slack %s %d | dd of=%s "

"oflag=seek\_bytes count=1 bs=%d seek=%d conv=notrunc",

cover\_file,

frag\_size,

tmp\_file,

block\_size,

skip\_size);

The frags that are being created as different from the size we are specifying in the code.

The strip\_null and restore\_null functions are working fine (as standalone, I tried them as separate C files and verified the output). Originally, the strip\_null function was leaving one extrabyte of data in the .dmg\_stripped file.

The .dmg\_retrieved file is currently not getting created, I mean file is created but zero bytes in size.

The .dmg\_restored is double the size of the original .dmg file. I have verified this with different sized .dmg files.

Expected outcome:

.dmg and .dmg\_restored should be of same size and content (hash verify)

.dmg\_stripped and .dmg\_retrieved should be of same size and content (hash verify)

The fragment size in dc->frag\_size is different from the frag\_size in os.path.getsize('./frags/frag.#') with # being one of the frag numbers.

In file\_encode(), when I printed out orig\_data\_size it was always 201. has

Also, the the frag\_size in os.path.getsize('./frags/frag.#') changes for some of the backends. See below:

For k=16, m=2:

JERASURE\_RS\_VAND frag\_size is 150bytes, JERASURE\_RS\_CAUCHY frag\_size is 8272bytes, BACKEND\_NULL frag\_size is 152bytes

For k=16, m=8:

JERASURE\_RS\_VAND frag\_size is 98bytes, JERASURE\_RS\_CAUCHY frag\_size is 8272bytes, BACKEND\_NULL frag\_size is 100bytes