Demo EncryptFS

Starting the Filesystem at ~/hello with Cloud backup and restore support

```
(base) kaypee@kaush:/mnt/c/Users/kaush/Documents/GitHub/AppliedCryptography/encryptedFS-WMerkelProofs$ make clean
rm -f encryptFS.out
rm -rf *.bin
# -rm -rf merkle_*.txt
(base) kaypee@kaush:/mnt/c/Users/kaush/Documents/GitHub/AppliedCryptography/encryptedFS-WMerkelProofs$ make
gcc -Wall -I./include -D_FILE_OFFSET_BITS=64 `pkg-config --cflags fuse openssl libsodium libcurl` -DFUSE_USE_VERSION=30 main.c ./src/fs_operation
s.c ./src/bitmap.c ./src/inode.c ./src/volume.c ./src/merkle.c ./src/crypto.c  ./src/cloud_storage.c -o encryptFS.out `pkg-config --libs fuse ope
nssl libsodium libcurl`
(base) kaypee@kaush:/mnt/c/Users/kaush/Documents/GitHub/AppliedCryptography/encryptedFS-WMerkelProofs$ make keygen
./encryptFS.out keygen ./key.txt
main: starting the file system
argc 3
argv[0] ./encryptFS.out
argv[1] keygen
argv[2] ./key.txt
Key stored in ./key.txt
(base) kaypee@kaush:/mnt/c/Users/kaush/Documents/GitHub/AppliedCryptography/encryptedFS-WMerkelProofs$ ./encryptFS.out -f -d ~/hello remote:encry
ptfs/superblock.bin ./key.txt
```

File create, read, write. (operation)

```
(base) kaypee@kaush:~/hello$ touch cat.txt
touch: setting times of 'cat.txt': Function not implemented
(base) kaypee@kaush:~/hello$ echo "hello i am cat" > cat.txt
(base) kaypee@kaush:~/hello$ cat cat.txt
hello i am cat
(base) kaypee@kaush:~/hello$
```

```
(base) kaypee@kaush:~/hello$ touch cat2.txt
touch: setting times of 'cat2.txt': Function not implemented
(base) kaypee@kaush:~/hello$ echo "hello" > cat2.txt
(base) kaypee@kaush:~/hello$ cat cat2.txt
hello
(base) kaypee@kaush:~/hello$ ls
cat.txt  cat2.txt
(base) kaypee@kaush:~/hello$ rm -rf cat2.txt
(base) kaypee@kaush:~/hello$ ls
cat.txt
```

File create, read, write. (file system debug)

Create

```
getattr /cat.txt
fs_op: getattr
fs_op: path: /cat.txt
inode: Reading inode 0
inode: Reading inode index in volume 0
   unique: 12, error: -2 (No such file or directory), outsize: 16
unique: 14, opcode: CREATE (35), nodeid: 1, insize: 64, pid: 22416
create flags: 0x8841 /cat.txt 0100644 umask=0022
fs_op: in create
bitmap: Reading bitmap for 0
inode: Allocating inode bitmap for 0
bitmap: Checking if bit 1 is free
bitmap: Setting bit 1
bitmap: Writing bitmap for 0
fs_op: inode_index after volume adjust: 1
inode: Writing inode 1
inode: Writing inode in volume 0
inode: Writing inode 1
key 5*L[+DSya29.a0AXooCgur4dKEiB1y4HMlO-uOaOc9P
N96ohXbENqZRdOvgiE2XMHi2uhk3LzT57H-B__33OQ-IARLyAsd0Xdqx9aCgYKARwSA
retFlag 0
inode: Reading inode 0
inode: Reading inode index in volume 0
inode: Writing inode 0
inode: Writing inode in volume 0
inode: Writing inode 0
```

A New Inode is allocated for the file, inode is encrypted and written to the inode file. Root dir inode is read decrypted, modified with a new child, and encrypted and written.

Read (Reading decrypted data and verifying integrity)

```
volume: Reading block 0
volume: Reading block 0 with volume 0
merkle: Block Hash: 88e4038cae5eefdd9dfb57213ec06dbadd197ab5489a200f2f7e41b06a701933
merkle: Decrypted hash: 88e4038cae5eefdd9dfb57213ec06dbadd197ab5489a200f2f7e41b06a701933
merkle: Leaf node: 88e4038cae5eefdd9dfb57213ec06dbadd197ab5489a200f2f7e41b06a701933
merkle: Verifying merkle path
merkle: Computed root hash: e064b80560aacb974dac7d6097dc6fb921b84c10f610f88288f1c3050e1057a4
merkle: Expected root hash: e064b80560aacb974dac7d6097dc6fb921b84c10f610f88288f1c3050e1057a4
merkle: Root hash matches -> Verified
```

With each block read the hashes and Merkle tree path is verified until the root.

Write (Storing encrypted and updating Merkle tree)

```
fs_op: write: new_block_index: 0
fs_op: write: volume_id_datablocks: 0
volume: Writing block 0
volume: Writing block 0 with volume 0
```

```
merkle: Lead node block index 0
merkle: New hash: 88e4038cae5eefdd9dfb57213ec06dbad
merkle: Updating merkle node
merkle: updated Node hash: 88e4038cae5eefdd9dfb5721
merkle: Hash comparison result: 1
merkle: Hashes match (stored and computed for leaf)
merkle: Saving merkle tree to file
merkle: Saving merkle tree to file
merkle: File path: ./merkle_0.bin
Merkle tree saved to file
fs_op: write: file_inode.size: 15
inode: Writing inode 1
inode: Writing inode in volume 0
inode: Writing inode 1
key 5[illegible]Sya29.a0AXooCgur4dKEi
N96ohXbENqZRdOvgiE2XMHi2uhk3LzT57H-B__33OQ-IARLyAsd
```

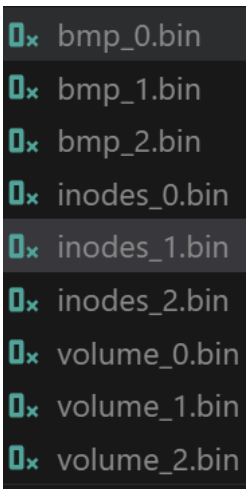
For every block we recompute the hashes on the Merkle path to root

Dynamic Volume Expansion (if the file needs more than what the current file system is allocated)

We create a big file which is more than what one volume can hold and observe that the file system expands by creating more files

```
(base) kaypee@kaush:~/hello$ cp ../cat.jpg ./
(base) kaypee@kaush:~/hello$ ls
cat.jpg  cat.txt
(base) kaypee@kaush:~/hello$ ls -lh cat.jpg
-rwxr-xr-x 1 kaypee kaypee 78K May  2 23:20 cat.jpg
(base) kaypee@kaush:~/hello$ feh cat.jpg
```

Newly created files




```
fs_op: dynamic_alloc: inode_index: -1
fs_op: dynamic_alloc: volume_id_new: 1
fs_op: dynamic_alloc: Expanding volume search
fs_op: dynamic_alloc: volume_num: 2
bitmap: Reading bitmap for 2
```

Cloud Backup on Unmount

```
cloud_storage: Uploading file bmp_0.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file inodes_0.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file volume_0.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file merkle_0.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file bmp_1.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file inodes_1.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file volume_1.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file merkle_1.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file bmp_2.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file inodes_2.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file volume_2.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file merkle_2.bin to folder encryptfs
File None created successfully.cloud_storage: Uploading file superblock.bin to folder encryptfs
File None created successfully.(base) kaypee@kaush:/mnt/c/Users/kaush/Documents/GitHub/AppliedCr
```

bmp_0.bin	me	23:30 me	8 KB
bmp_1.bin	me	23:30 me	8 KB
bmp_2.bin	me	23:30 me	8 KB
inodes_0.bin	me	23:30 me	20 KB
inodes_1.bin	me	23:30 me	10 KB
inodes_2.bin	me	23:30 me	0 bytes
merkle_0.bin	me	23:30 me	3 KB
merkle_1.bin	me	23:30 me	3 KB
merkle_2.bin	me	23:30 me	3 KB
superblock.bin	me	23:30 me	10 KB
volume_0.bin	me	23:30 me	81 KB
volume_1.bin	me	23:30 me	81 KB
volume_2.bin	me	23:30 me	24 KB

All volume-associated files and superblocks are backed up on Google Drive.

Mounting from cloud files

```
(base) kaypee@kaush:/mnt/c/Users/kaush/Documents/GitHub/AppliedCryptography/encryptedFS-WMerkelProofs$ make clean
rm -f encryptFS.out
rm -rf *.bin
# -rm -rf merkle_*.txt
(base) kaypee@kaush:/mnt/c/Users/kaush/Documents/GitHub/AppliedCryptography/encryptedFS-WMerkelProofs$ make
gcc -Wall -I./include -D_FILE_OFFSET_BITS=64 `pkg-config --cflags fuse openssl libsodium libcurl` -DFUSE_USE_VERSION=30 ma
in.c ./src/fs_operations.c ./src/bitmap.c ./src/inode.c ./src/volume.c ./src/merkle.c ./src/crypto.c  ./src/cloud_storage.
c -o encryptFS.out `pkg-config --libs fuse openssl libsodium libcurl`
(base) kaypee@kaush:/mnt/c/Users/kaush/Documents/GitHub/AppliedCryptography/encryptedFS-WMerkelProofs$ ./encryptFS.out -f
-d ~/hello remote:encryptfs/superblock.bin ./key.txt
```

```
volume: Downloading superblock from remote storage
volume: Directory: encryptfs
volume: Filename: superblock.bin
cloud_storage: Downloading file superblock.bin from folder encryptfs
volume: Superblock loaded
cloud_storage: Downloading file inodes_0.bin from folder encryptfs
cloud_storage: Downloading file bmp_0.bin from folder encryptfs
cloud_storage: Downloading file volume_0.bin from folder encryptfs
cloud_storage: Downloading file merkle_0.bin from folder encryptfs
cloud_storage: Downloading file inodes_1.bin from folder encryptfs
cloud_storage: Downloading file bmp_1.bin from folder encryptfs
cloud_storage: Downloading file volume_1.bin from folder encryptfs
cloud_storage: Downloading file merkle_1.bin from folder encryptfs
cloud_storage: Downloading file inodes_2.bin from folder encryptfs
cloud_storage: Downloading file bmp_2.bin from folder encryptfs
cloud_storage: Downloading file volume_2.bin from folder encryptfs
cloud_storage: Downloading file merkle_2.bin from folder encryptfs
```

Tampering with a block in volume to see decryption fail

Before tampering

```
(base) kaypee@kaush:~/hello$ echo "hello i am a very cyte cat heheheh sdklfgjhasdjklfg asjkhlgjklsadhfg jksdhfjkshadf hsdf
lksjadhf lskdhjflkasjdhfjashf sadhjgfksgf sfghasfhjgsjaklhf saljkfhasdjkhf lasjdfhjklasfhasdkljhf skladjhfsdjkahfjkshadfga
sfhjkgasdlfhjkasdghlfjk shadjfhsdjafhlasdjkfh sadjkfhlsjkadf hsjkladhf jklashdf kljhasdlf kjhsdaklf jsdhafklj" > cat.txt
(base) kaypee@kaush:~/hello$ cat cat.txt
hello i am a very cyte cat heheheh sdklfgjhasdjklfg asjkhlgjklsadhfg jksdhfjkshadf hsdflksjadhf lskdhjflkasjdhfjashf sadhj
gfksgf sfghasfhjgsjaklhf saljkfhasdjkhf lasjdfhjklasfhasdkljhf skladjhfsdjkahfjkshadfgasfhjkgasdlfhjkasdghlfjk shadjfhsdja
fhlasdjkfh sadjkfhlsjkadf hsjkladhf jklashdf kljhasdlf kjhsdaklf jsdhafklj
```

Tampering

Before

```
volume_0.bin
39    N?|V#?&?t?~n??NAK?\?)<g???2
```

Changing the last byte we get

After

```
volume_0.bin
39    N?|V#?&?t?~n??NAK?\?)<g???1
```

.After Tampering

```
(base) kaypee@kaush:~/hello$ cat cat.txt
(base) kaypee@kaush:~/hello$
```

```
merkle: Getting merkle tree for volume 0
merkle: Volume tree: 0x5579cad7ca50
merkle: Expected root hash: b0758af7abd62dc9c4ee22ac4d49b1678b0dd3f992e1132838620ace94f9ff2e
merkle: Getting block hash
volume: Reading block 0
volume: Reading block 0 with volume 0
volume: Decryption failed for block 0 in volume 0
merkle: Block Hash: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
merkle: Decrypted hash: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
merkle: Leaf node: 9c360edfa67b7565c331caa418a29072f8afac1d2022dedc4234b0c82760903e
merkle: Verifying merkle path
merkle: Computed root hash: bdca9e8dbca354e824e67bfe1533fa4a238b9ea832f23fb4271ebeb3a5a8f720
merkle: Expected root hash: b0758af7abd62dc9c4ee22ac4d49b1678b0dd3f992e1132838620ace94f9ff2e
merkle: Root hash does not match -> Not Verified
volume: Integrity check failed for block 0 in volume 0
```