Group 6 Fuse Project

HS, CS, SW, Brandi McWilliams

Abstract - Our group collaborated to create a FUSE filesystem integrating a XOR encryption function for reading and writing files.

Introduction

The following report details the planning, design, and implementation of a FUSE filesystem with encryption using Python.

Planning

Our group decided to use Discord to work on the project and pitch ideas of how to proceed successfully. We met on a few occasions during class when time allowed. We agreed to work together on Saturdays at 5:00 p.m. Our group began by implementing the FUSE filesystem provided by Stavros. Once we were able to complete the basics of the passthrough in Python, our team began working on the encryption function for the project.

Design

Using the python code from Stavros as our foundation, we implemented a XOR encryption function utilizing a key, provided by the user, appended to the end of a filename. This required modifying a few of the original functions as well as providing the XOR function to encrypt and decrypt the files.

Implementation

Three of the original functions were modified to implement the encryption. They were the following:

A. *_full_path*

The _full_path function was modified to search out the key contained within the filename.

```
def _full_path(self, partial):
        index = partial.rfind("_")
# find the position of the key
        if (index == (len(partial) -
2) and (len(partial) > 1)):
             self.key =
partial.split("_")[-1]
             partial =
partial[:index]
# find the file name
        else:
            self.key = ''
# initialize key if there is no one
         partial =
partial.lstrip("/")
        path =
os.path.join(self.root, partial)
        return path
```

B. *read*

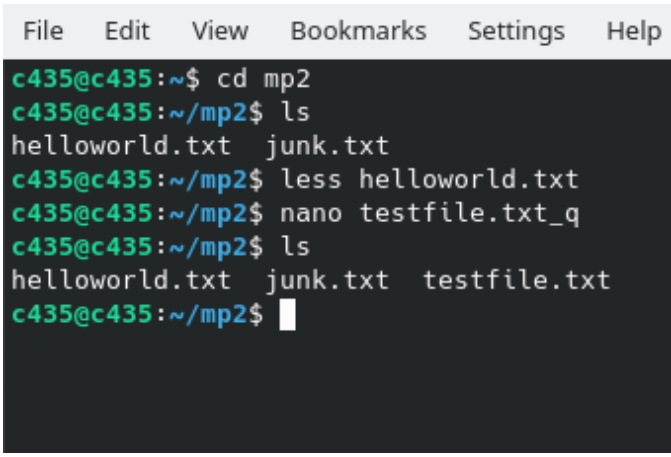
The read function implements the encryption function using the key, to decrypt the file read.

```
def read(self, path, length, offset,
fh):
        os.lseek(fh, offset,
os.SEEK_SET)
        re = os.read(fh,length)
# read the data from the file first
        decr = self.encry(self.key,
re)
# use encryption algorithm to
decrypt the data
        return decr
```

C. *write*

The write function implements the encryption function to XOR the data before writing the file.

```
def write(self, path, buf, offset,
fh):
        os.lseek(fh, offset,
os.SEEK_SET)
        enbuf = self.encry(self.key,
buf)      # use encryption algorithm
to encrypt the data before write
into disk
        return os.write(fh, enbuf)
```



(Above: Image displaying the creation of the testfile with key q.)

```
File   Edit   View   Bookmarks   Settings   Help
c435@c435:~/mp2$ ls
helloworld.txt  junk.txt  testfile.txt
c435@c435:~/mp2$ cat testfile.txt
%QQQQQLQ_{c435@c435:~/mp2$ cat testfile.txt_
This is testing with key = q.
c435@c435:~/mp2$
```

(Above: Image comparing encryption if the key isn't included and output when key is included in filename by user.)

D. *encry*

The encryption function uses a conditional to check for the key (integers or characters). If there is no key attached to the filename, a key is 'assigned' as '0'. The data gets placed into an array where the XOR can be performed on each index. The data is then cast as a string to be returned.

```
def encry(self, key, data):
        if(key == ''):
            key_int =0
# if no key exists, use 0 to encrypt
        else:
            key_int = ord(key)
# change the key character into  an
integer
        stringline = ''
        byte_array =
array.array('B', data)    # read the
data in bytes and store in an array
        for onebyte in byte_array:
          encrypt = onebyte ^
key_int          # use exclusive-or
to encrypt/decrypt
          stringline += chr(encrypt)
# store the data in a string
        return
bytes(stringline,'utf8')
```

```
File   Edit   View   Bookmarks   Settings   Help
c435@c435:~/mp2$ nano bones.txt_3
c435@c435:~/mp2$ ls
bones.txt  helloworld.txt  junk.txt  testfile.txt
c435@c435:~/mp2$ cat bones.txt
zP[\\@VG[VWR]TVA9gV@GXVJZ]GVTVA]F^QVA9c435@c435:~/mp2$ cat bones.tx
I choose the danger.
Test key integer number 3.
c435@c435:~/mp2$
```

(Above: Another testing example that uses a number for the key.)

Sources

Korokithakis, Stavros. *Writing a FUSE filesystem in Python.* Stavros' Stuff, October 2013, https://www.stavros.io/posts/python-fuse-filesystem/. Accessed 14 Apr 2022.