# Silent Assassins - Secure Personal Cloud

170050104(33%), 170050106(34%), 170050107(33%)

November 24, 2018

**Declaration**

I acknowledge and understand that plagiarism is wrong. This project is my own work, or my group's own unique group project. I acknowledge that copying some- one else's work, or part of it, is wrong, and that submitting identical work to others constitutes a form of plagiarism.

## 1 Work Done till Mid-term

We created a web-client which supports signup and login for users. It also displays all the files uploaded by the user and allow them to download. We implemented upload through linux client and web client, and implemented copying of the directory structure to the database on the server.

## 2 Sync and Version Control

We implemented this as a part of linux client. First, we implemented a bash function for version control and then used it for implementation of sync. In the case of files with same name but different content our function takes the opinion of the client whether to trust the server's files or his folder's files and works accordingly. Our sync function takes care of version control along with the syncing.

## 3 Encryption on linux client end

We provide the users with 3 different options of encryption schemas(AES, DES3, and Blowfish). While uploading a file through the linux client, the file first gets encrypted according to the choice of key and encryption schema and then is saved to the database on the server. We give the user liberty to shift to a different encryption schema at any point of time through the linux client. If this is done then the files on the server database would get updated according to the new encryption schema.

## 4 Web-end Decryption and File Rendering

On the web-end, after logging in the user has to fill the private-key and encryption schema through an input stream. The fields entered are then stored in session storage(user has to input key each time the webpage is opened) and are used to decrypt the file on clicking. Clicking also leads to the file being displayed in a new window. This is done through the post() and get() methods in jQuery.

## 5 *File Sharing*

We have worked a lot on file sharing. We designed a new algorithm using ideas of AES and RSA that is quite secure and compatible for large file uploads. We made some assumptions regarding this algorithm. We gave every user 6 keys named as private_key , shared_key , schema , n, e, d where only n and e are public keys. Now lets say A wants to send a file to B then B will upload it's shared_key to the database after encrypting it with the n and e of A (RSA). Then A will decrypt the shared key of B using it's d and upload the file while encrypting it with shared key of B. B then downloads the files, decrypts them and encrypt

them again with it's private key. Here note that the private key d used for RSA of A is known to just A as it has been encrypted and then stored. Also the shared key of B is only passed on to A and not known by server or anyone else, hence protecting A's data while sharing. The major one of them being that if we want to share file between A and B then both A and B should be online at that instance.

# 6 Websites used while coding


1)https://stackoverflow.com/questions/18367007/python-how-to-write-to-a-binary-file
2)https://stackoverflow.com/questions/35310695/convert-byte-string-to-bytes-or-bytearray
3)https://stackoverflow.com/questions/36631419/display-data-from-a-database-in-a-html-page-using-django
4)https://stackoverflow.com/questions/12517451/automatically-creating-directories-with-file-output
5)https://stackoverflow.com/questions/3431825/generating-an-md5-checksum-of-a-file
6)https://gist.github.com/caiguanhao/9446527
7)https://stackoverflow.com/questions/29512858/cryptojs-and-key-iv-length
8)https://github.com/agorlov/javascript-blowfish
9)https://www.youtube.com/user/thenewboston/playlists
10)https://wsvincent.com/django-user-authentication-tutorial-login-and-logout/
11)https://github.com/wsvincent/django-auth-tutorial
12)https://piazza.com/class/jmg8gtyj64u35n?cid=44
13)https://github.com/buckyroberts/Viberr/blob/master/music/views.py
14)https://www.w3schools.com/sql/sql_datatypes.asp
15)https://sqliteonline.com/
16)https://curl.haxx.se/docs/manual.html
17)https://djangobook.com/authentication-web-requests/
18)https://docs.djangoproject.com/en/2.0/releases/1.10/
19)https://groups.google.com/forum/#!topic/django-users/pQbRxuIeOzE
20)https://simpleisbetterthancomplex.com/tips/2016/05/05/django-tip-1-redirect.html
21)https://www.bogotobogo.com/python/Django/Python_Django_Image_Files_Uploading_Example.php
22)https://github.com/mbraak/django-file-form/issues/111
https://groups.google.com/forum/#!topic/django-users/VfAJSj8gVI8
23)https://stackoverflow.com/questions/1078897/storing-file-content-in-db
24)https://stackoverflow.com/questions/4028904/how-to-get-the-home-directory-in-python