CS628A : Computer System Security Assignment 1

Deepesh Chaudhari (19111028) Lavlesh Mishra (19111048)

August 2019

1 User Struct

The user stuct contains username, password, RSA private key pair and a map: Myfiless (which maps a file to a **Filemetadata** Struct), HMAC, a list of keys used to encrypt each block of file and a list of address of all the file blocks and number of blocks. Filemetadata contains the metadata of file which includes two maps namely ”blocks” and ”blockMac”, EncryptKey and size of the file.

2 InitUser(username string, password string)

To save the user information, generate RSA key pair and populate the user data structure, generate a key ’K’ using **Argon2key** on password and using username as salt. Then we encrypt (with key K) using **CFBEncrypter** and HMAC the User Struct and save it at location userID. where userID is UUID (HMAC (”username”)) and register the RSA public key to the key store using the username.

3 GetUser(username string, password string)

To get the user information we generate key K using username and password, then regenerate the userID and load from datastore, check the HMAC and decrypt.

4 StoreFile(filename string, data []byte)

To store a file we generate **Filemetadata**, store it in user’s Myfiles map. This also need us to calculate the size of the file (check if its a multiple of blocksize) and divide it into blocks(of blocksize) and calculate address for each block as addr = string(Hash(userlib.RandomBytes(32))) and create encryption key as E := userlib.Argon2Key([]byte(user.Password), Hash([]byte(user.Username)), 16) and store the values in the Filemetadata. Then we encrypt the file using CF-BEncrypter, having ’E’ as key and calaculate HMAC and store in Filemetadata.

5 LoadFile(filename, blockOffset integer)

To load a file for a particular blockoffset, load the file by calculating 'addr' and 'E' in the same manner as done in storefile. Then fetch decrypt and verify and access.

6 AppendFile(filename string, data []byte)

Load the file metadata using filename and load the file, verify if the file is not corrupted and check if the size of the value is a multiple of blocksize. After it store the value and update the entries in the **Filemetadata**.

7 ShareFile()

To share a file we retrieve Filemetadata of the file from Myfiles map with in the sender's user struct. Then retrieve the receiver's public RSA key from keystore annd use this to encrypt the metadata and return encrypted message and signature.

8 Receive_Share()

To receive a shared file under name F_i we get the sender's public key from the keystore, verify the RSA signature, then use the receiver's private key to decrypt the data. We then save this data in the Myfile map in the receiver's User Struct under name F_i.

9 RevokeFile()

To revoke access to a shared file named f, we first check that f is contained in the MyFiles map of the users's struct. We then copy the filedata in a temporary storage, Load the file and delete all the ledata and metadata, then use StoreFile() to store it again in the datastore.