

Report

Secure Assignment Sharing

Implemented by Attribute based encryption

Note:

The given openABE library was not working. Thus another library for encryption was done, named as “**easy-encryption**”, the link of GitHub is as follow:

<https://github.com/philipperemy/easy-encryption>

Contents

Introduction:	3
• Attribute Universe	3
• Summary of Functionality	3
Screenshots of working:	4
Module Explanation:	12
1. Registration	12
2. Login	12
3. Attribute Authority	12
4. Key Management Authority	12
Revoke:	13
5. Storage Services	14
6. Encryption Decryption Services	14
Security Flaws:	14

Secure Assignment Sharing (SAS)

Introduction:

Secure Assignment is a platform for sharing the assignments securely. In this system(organization) you can create your assignment with the specific attributes associated with that assignment. As it's based on the concept of **attribute based encryption**. Thus each uploaded assignment have attributes specified to it and then user that qualifies that attributes are able to see the decrypted assignment.

- **Attribute Universe**

It can be defined as the domain of the attributes that a producer can set or attributes that a user profile can have. The attribute universe of **SAS** is:

```
string CGPA;  
string semesterNo;  
string batchNo;  
string section;  
string age;  
string gender;  
string department;  
string favSubj;  
string inchargeTeacher;
```

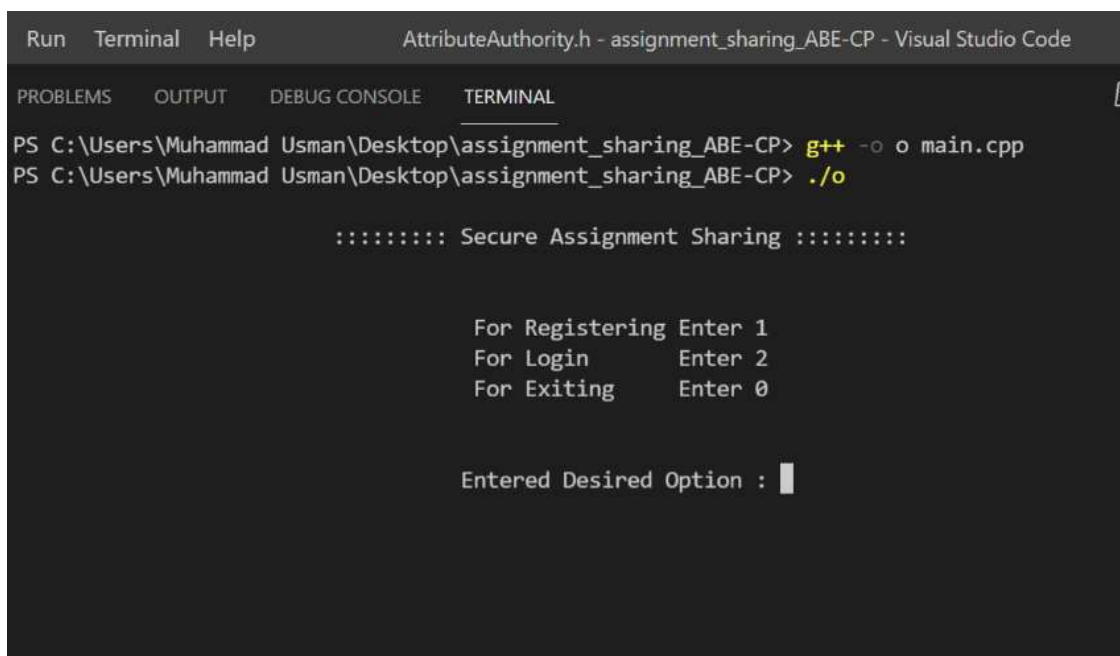
Each of the above is the attribute regarding to the assignment. These all make the attribute universe.

- **Summary of Functionality**

- In start a register/login menu was displayed by which a user can register or login according to need. Data for the registered user is being stored in a text file “**register.txt**”, contains the whole registered data with the profile, that contains attribute associated to them. At the time of registration attributes are taken by the **Attribute Authority**. And Keys are generated accordingly by **Key Management Authority**.

- Then different function menu is given to perform the functions, from which user can upload an assignment and can specifies the attribute universe and is associated to the assignment and all of this is being encrypted by **Encryption Decryption Services** with the help of keys and is stored by the **Storage Services**.
- User can login and can also logout anytime. The list of all assignments available is also available from there using a unique code user can see the files and can have access to decrypt if attributes matches with the cipher Text policy.
- Users can also revoke the information associated with the assignment file. But the user must be owner of that file. By revoking mean user if owner is allowed to change the attributes associated to that assignment at any time.

Screenshots of working:



```
Run Terminal Help AttributeAuthority.h - assignment_sharing_ABE-CP - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\Muhammad Usman\Desktop\assignment_sharing_ABE-CP> g++ -o o main.cpp
PS C:\Users\Muhammad Usman\Desktop\assignment_sharing_ABE-CP> ./o

::: Secure Assignment Sharing :::

For Registering Enter 1
For Login Enter 2
For Exiting Enter 0

Entered Desired Option : 
```

```

      For Login      Enter 2
      For Exiting    Enter 0

      Entered Desired Option : 1

      Enter Name : asghar
      Enter user-name : asghar_3
      Enter password : ****

      :::::::::: Setup Your profile ::::::::::

      Note : If you don't want to enter any attribute enter NULL
      But remember incomplete file will let you to face problems in accessing the material :)

      CGPA : 23
      Semester No : 2
      Batch No : 20
      Section : 10
      Age : 23
      Gender : null
      Department : null
      Favourite Subject : null
      Class incharge : ali

      ---- Registered Successfull :::: logging you ----

```

```

      :::::::::: Welcome asghar_3 ::::::::::

      :::: Secure Assignment Sharing provides you following ::::

      Enter 1 to see list of documents uploaded in SAS
      Enter 2 to upload your assignment in SAS
      Enter 3 to redifine (revoke) attributes of your assignment in SAS
      Enter 0 to Logout from SAS
      Enter any other character to access assignment in SAS

      Enter Your desired option : █

```

Login and registration

```

##### welcome assign_3 #####

::: Secure Assignment Sharing provides you following :::

Enter 1 to see list of documents uploaded in SAS
Enter 2 to upload your assignment in SAS
Enter 3 to redefine (revoke) attributes of your assignment in SAS
Enter 0 to Logout from SAS
Enter any other character to access assignment in SAS

Enter Your desired option : 1

##### SAS have the following Assignments in storage #####

Assignment ID : Assignment Name

2255 : coal.txt
2244 : oop.txt
433 : e.txt
1122 : networks.txt
2245 : network2

Press any key to continue . . .
```

Showing the assignments in the storage

```

Assignment ID : Assignment Name

2255 : coal.txt
2244 : oop.txt
433 : e.txt
1122 : networks.txt
2245 : network2

Press any key to continue . . .

Enter the assignment code u want to see : 2245

##### Access Denied #####

##### Encrypted File is #####

iikU0lZëhfY e*_Z!#h#o
```

Attributes were not matched the encrypted message shown

```

      ::::: Creating file of nexali.txt :::::

Enter data : this is a file that will be save encryptedly

      ::::: Enter the attributes you want to specify with the file :::::
      ::::: Something not allowed can be done by entering NULL :::::

      CGPA : null
      Semester No : null
      Batch No : null
      Section : null
      Age : null
      Gender : null
      Department : null
      Favourite Subject : null
      Class incharge : null

      ::::: Saving File :::::

      ::::: Encrypted form of nexali.txt is saved in storage :::::

      ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

Enter 1 to see main menu again
Enter 0 to logout

Enter desired option : █
```

A file is created that can be accessed by anyone as no attribute was specified all were null. In below screenshot the owner was allowed as shown below, then will logout and will access with another user as shown in below screenshots.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL [g++ + v]
```

```
Enter Your desired option : 1

:::::::::: SAS have the following Assignments in storage ::::::::::

Assignment ID   : Assignment Name
2255            : coal.txt
2244            : oop.txt
433             : e.txt
1122            : networs.txt
2245            : network2
1980            : nexali.txt

Press any key to continue . . .

Enter the assignment code u want to see : 1980

::::: You are the owner of this file ::::::

:::::::::: Decrypted File is ::::::::::

this is a file that will be save encryptedly
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
```

```
:::::::::: Secure Assignment Sharing ::::::::::

For Registering Enter 1
For Login       Enter 2
For Exiting     Enter 0

Entered Desired Option : 2

Enter The credentials to Login

User-Name : user1
Enter password : ****
```



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  [x]

Enter Your desired option : 1

:::::::::: SAS have the following Assignments in storage ::::::::::

Assignment ID   :   Assignment Name
      2255      :   coal.txt
      2244      :   oop.txt
       433      :   e.txt
      1122      :   networks.txt
      2245      :   network2
      1980      :   nexali.txt

Press any key to continue . . .

Enter the assignment code u want to see : 1980

:::::::::: Accessed Successfully ::::::::::

:::::::::: Decrypted File is  ::::::::::

this is a file that will be save encryptedly
```

Now revoking the file attributes that were created and setting that the gender is f that user 3 have then will be accessed by user3 but not by user1. In some cases, the encryption is not too much secure because of the library used.

```
Assignment ID : Assignment Name
2255 : coal.txt
2244 : oop.txt
433 : e.txt
1122 : networks.txt
2245 : network2
1980 : nexali.txt

Press any key to continue . . .

Enter id of the assignment : 1980

Enter the new attributes for file

CGPA : null
Semester No : null
Batch No : null
Section : null
Age : null
Gender : f
Department : null
Favourite Subject : null
Class incharge : null

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
```

```
:::::::::::: SAS have the following Assignments in storage ::::::::::

Assignment ID : Assignment Name
2255 : coal.txt
2244 : oop.txt
433 : e.txt
1122 : networks.txt
2245 : network2
1980 : nexali.txt

Press any key to continue . . .

Enter the assignment code u want to see : 1980

:::::::::::: Access Denied :::::::::::

:::::::::::: Encrypted File is :::::::::::

this is a file that will be save ec#ryptedly
```

```
2255 : coal.txt
2244 : oop.txt
433 : e.txt
1122 : networs.txt
2245 : network2
1980 : nexali.txt
```

Press any key to continue . . .

Enter the assignment code u want to see : 1980

:::::::: Accessed Successfully ::::::::::

:::::::: Decrypted File is ::::::::::

this is a file that will be save encryptedly

Press any key to continue . . .

::::: Secure Assignment Sharing provides you following :::::

Enter 1 to see list of documents uploaded in SAS
Enter 2 to upload your assignment in SAS
Enter 3 to redifine (revoke) attributes of your assignment in SAS
Enter 0 to Logout from SAS
Enter any other character to access assignment in SAS

Enter Your desired option : 0

::::::::: See you soon user3 :) ::::::::::

Logout the user3 and all like that.

Module Explanation:

1. Registration

This helps in the process of registration of the new users. Mainly it takes the credentials and then write that credentials to the storage that later helps on for the logging in of the user. Thus registration Module main goal is to register the user. Main thing that registration do is take the attributes, profile setup of user that is further stored and is used in different things.

2. Login

This helps in the login of the user. If the user is registered that user is logged in. Login credentials are the user name and password. This is also acting like the thing that keep the track of the user. In such a manner that it maintains which user is currently logged in. When user Logs out the session is deleted and indicates no user at that time.

Both of the above have the implementation of password hiding. They hide the entered password. Furthermore, uses the attribute universe and attribute authority to set the attributes and also uses the store and other services.

3. Attribute Authority

This authority is responsible for maintain the attributes. This module has different functions, two of the important are to collaborate with the storage services and read and write the attributes for a specific file or for a specific id. Attribute Universe is also the responsibility of the attribute authority. Thus attribute authority has the complete authority on attributes and helps in the maintaining the attributes.

4. Key Management Authority

This module has one of the most important work to perform. KMA manages the key that are being develop and generate the different keys. The three different keys generated are:

- Company key
- User's key
- Cipher text policy key

Company key is just made to have more secure system. This key specifically is generated by the organization and keeps its data encrypted with this.

User's key is the key that is made for each user and is run time. When user request to access some material the key is generated on the basis of attributes.

Cipher text policy key is the key that is being associated with the policy by which document is going to be decrypted. The attributes. At the runtime when the file is made this key is generated and stored in the database.

First of all, when program starts the **KMA** generates a key that is said to be the company key and is stored in the module. After that user comes in explore things and when he tries to upload a document the attributes are taken. And on the basis of that a cypher policy is defined. Here attributes are added with ">" then it is encrypted by the lib included, with the company key. Then is saved in the file that a file associated to this will be decrypted only by this key. When an owner wants to access the attributes are not checked but the access is given as he is the author of that. When user wants to access the key the user key is generated and then is compared with the cipher text policy if they are equal than the decrypted assignment is encrypted by the user's key. If it was entered that file can be accessed by a female and all other are don't care as in the above example. When user try to access such file, a dummy user is created in which all other attributes become null except the gender is gender is same as required then the user key developed will be same. Thus will be used in the decryption of that encrypted data. The key directly is not used as in encryption process the key was encrypted by the company's key similarly the user key first developed s encrypted with company's key and then is used for decryption. Encryption and decryption is done by the library included. Mentioned above.

Revoke:

The other main option **KMA** performs is the revoke operation. The revoke operation done is on the basis of the attributes. A user if wants to revoke change the attributes, he must be the owner of that file. If user is not the owner, he is not given access to revoke.

The revoke process follows that id for the file is taken for which attributes are to be changed then according to older cipher policy the data is encrypted to original one and is saved. Then new attributes are taken and the new cipher policy and key is generated as discussed above. Then data is encrypted by that cipher policy. The main goal was writing that attributes in the memory. Two files were used for this purpose from one the data was read and was pasted to other and the lines were skipped if the id matches. Thus first the data was moved from the files then the new attribute, new data for file was appended in the storage. Thus by the updating in memory all other things were done automatically and were changed automatically.

All of the modules were used in this, Attribute Authority was used to handle the attributes, KMA was used to handle keys, the login user's data was used, the Storage service was also used to read and write data in the memory and was encrypted and decrypted by the Encryption and Decryption services.

5. Storage Services

This module was used to store the data in the back end. This was just the cloud storing service. It has different functions. Attributes were set and were accessed. Furthermore, to access a specify id, cipher policy or any data regarding to the file was done by the storage service. Mainly storage service was responsible of storing and reading the data and was mainly used in each module. The hypothetical storage was made by using files.

6. Encryption Decryption Services

This module was used for the encryption and decryption processes by using the library included. All type of encryption and decryption was done by it. The whole file encryption was done by reading whole file as a single string then that string was encrypted by using the keys and then stored to the file. Similarly, the whole encrypted file was read and was move to a single string that was decrypted but was not stored anywhere, was only visible to the right ones on the terminal. The wrong messages were stored after taking data to mislead the attacker.

Security Flaws:

Secure Assignment Sharing has many security flaws and strengths, some of them are as follow:

Attribute Authority has given the full authority on the attributes as the attributes are the most important thing that should be secured as all of the process is based on them. All the things are encrypted on the basis of attributes. The first threat to giving the whole access to attribute authority is if the server is being attacked. Thus attribute authority should have made more secure as in our implementation this factors lack. The attributes access is direct and can also write in memory rather than reading, this is also a threat. This can be minimized by setting up attributes in some shuffled manner and making not totally dependent on the attributes.

Key management authority although creates the keys that are encrypted double by the company key, but if company key is attacked that will be a whole system destruction, we were unable as library was not working and lack of experience, the company key can be runtime and should be changed after a time and should be unique in each section that will be make him more secure. There threat and risk to our system because of same key each time, a static key. Storage services has also security flaws we are writing the data simply. A mechanism should be made so that the data store be also in the encrypted form. And all data is stored in the same file, same memory this should be spread to minimize the threats. Similarly, encryption and decryption is done by a simple library that in many cases not encrypt of that level.

During revoke process the old data is decrypted back to original and is then encrypted back and the whole storage is being manipulated. Thus the secure sharing is necessary. Insiders can also

leak the data in every case thus mechanism is needed that trust ones know while from others the data is hide. What they see should not be that 😊.

Encryption and decryption services are if attacked directly then after their work there is a garbage value in the labels of encrypted and decrypted to confuse the attacker. When the accessed is denied the encrypted data is encrypted more by random strings without saving and show that double encrypted so that a kiddy attacker is more puzzled and cannot figure out the key. Similarly, in revoke function first it is checked the user is owner of the file or not if user is not he is not allowed to revoke the file conditions as he is not the owner of the file.

If we talk generally there are many other flaws like any one can disguise himself after obtaining the id as storage is not secure. Thus with making the storage secure, identifying the correct user is also needed. SAS doesn't have any check on brute force if attributes are brute forced that may can found the one that can hack the organization. The most significant security strengths that we applied include that only an owner can change attributes but anyone can disguise. In addition to this the profile can be setup once, a user cannot change attributes, that may be initial step to minimize the brute force. Although there are security flaws that can be used to exploit the Secure Assignment Sharing (SAS).

