

SECURE FILE STORAGE IN HYBRID CLOUD

by

Praveen Ramani- pxr170005

Shivaprakash Sivagurunathan- sxs180104

DESCRIPTION:

Due to the increasing availability of fast and cheap computing power, it has become computationally feasible to break encryption. There have been reports that some hackers have used AWS cloud computing to break encryption. Increasing the key size seems to be the obvious solution, but the problem is that it requires more computation power to encrypt and decrypt the file which can lead to higher cost, affect response time and performance.

The proposed system provides a way to securely store and retrieve files in a hybrid cloud environment without increasing the key length.

WORKING:

Storing the files:

- The system splits the file into a random number of parts.
- Randomized encryption techniques are used to encrypt each file part using a randomly selected encryption algorithm such as Fernet and RC4.
- The encrypted parts are then given a unique ID.
- The unique ids and their corresponding cipher text are stored in the public cloud while the unique ids, encryption method and encryption keys are stored in the private cloud.

Retrieving the files:

- The system fetches the meta data of the required file, from the private cloud.
- Using the meta data, the system then fetches the encrypted file parts from the public cloud.
- Each file part is decrypted using their corresponding key and algorithm, which is got from the meta data.
- The decrypted files are merged in the correct order to get the original file.

TECHNOLOGIES USED:

- Randomized encryption algorithms
- Firebase real-time database (private cloud)
- Firebase storage bucket (public cloud)

ADVANTAGES:

- Does not require large encryption keys.
- Scalability.
- Security increases with scaling.
- Small private cloud, which decreases attack surface and cost.

CHALLENGES FACED:

- **Byte data Storage:**

Firestore real-time database does not allow the data to be stored in byte format.

- **Solution:**

- Convert the byte into a string and store it. Convert back into bytes after fetching.

- **RC4 key storage:**

Although the above solution worked for fernet keys, it did not work for RC4 keys because converting it back to bytes changes the size.

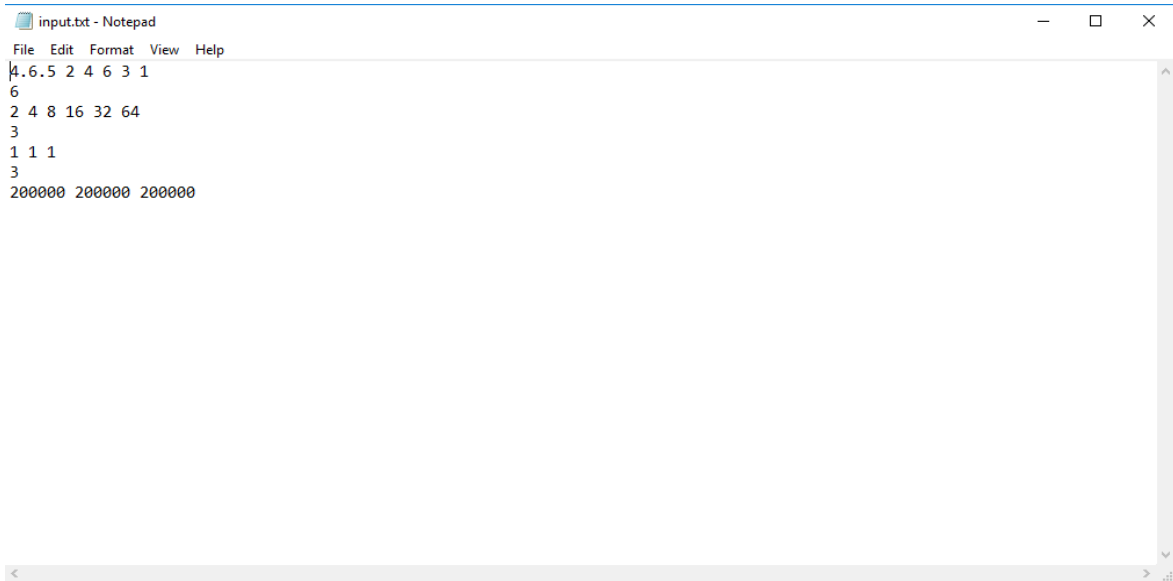
- **Solution:**

- Store the key in a separate file.
- Give a unique ID for this file and add it as a part of meta data.

SCREENSHOTS:

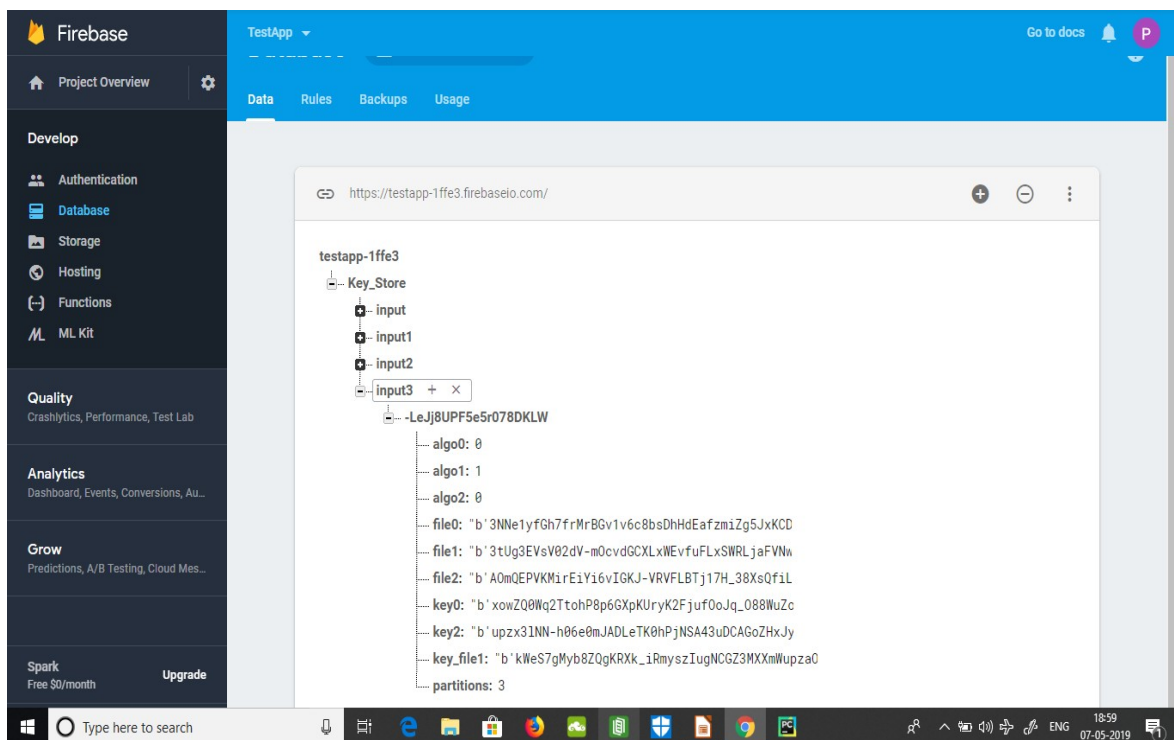
Input file:

The input file consists of a few line of data.



```
input.txt - Notepad
File Edit Format View Help
4.6.5 2 4 6 3 1
6
2 4 8 16 32 64
3
1 1 1
3
200000 200000 200000
```

Real-Time database (Private Cloud):

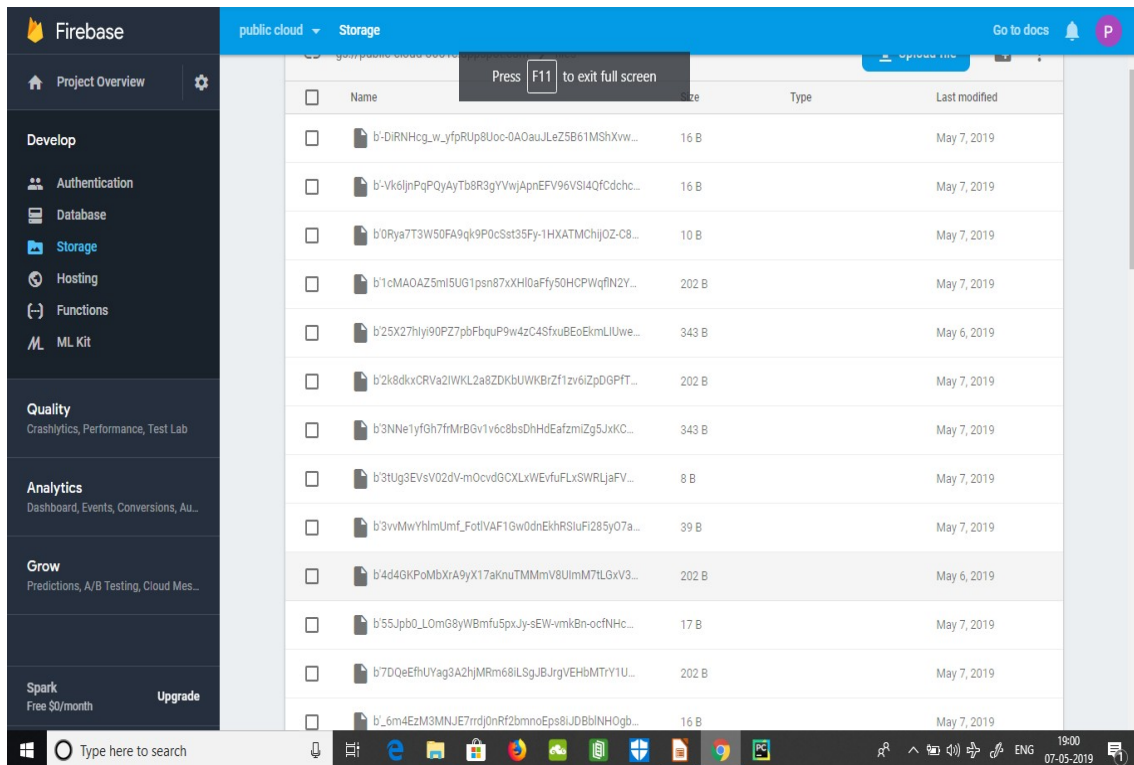


The screenshot shows the Firebase console interface. On the left is a sidebar with navigation options: Project Overview, Develop (Authentication, Database, Storage, Hosting, Functions, ML Kit), Quality (Crashlytics, Performance, Test Lab), Analytics (Dashboard, Events, Conversions, Au...), Grow (Predictions, A/B Testing, Cloud Mes...), and Spark (Free \$0/month, Upgrade). The main area displays the 'Data' tab for the 'testapp-1ffe3' database. The database structure is shown as a tree under 'Key_Store':

- Key_Store
 - input
 - input1
 - input2
 - input3
 - LeJ8UPF5e5r078DKLW
 - algo0: 0
 - algo1: 1
 - algo2: 0
 - file0: "b'3NNe1yfGh7frMrBGv1v6c8bsDhHdEafzmiZg5JxKCD"
 - file1: "b'3tUg3EVsV02dV-m0cvdGCXLxWEvfFLxSWRLjaFVNw"
 - file2: "b'A0mQEPVKM1rEiYi6vIGKJ-VRVFLBTj17H_38XsQf1L"
 - key0: "b'xowZQ0Wq2TtohP8p6GXpKuryK2Fjuf0oJq_088WuZc"
 - key2: "b'upzx31NN-h06e0mJADLeTK0hPjNSA43uDCAGoZHXJy"
 - key_file1: "b'kWeS7gMyb8ZQgKRXk_iRmyszIugNCGZ3MXmIupza0"
 - partitions: 3

- Contains the meta data.
- Here the file has been split into 3 parts, each having:
 - **file** : The unique ID of the file
 - **algo** : The code of the algorithm used to encrypt the file.
 - **Key** : The key used to encrypt and decrypt the file.
 - **key_file** : The unique ID of the file that contains the key. This is only if the file was encrypted using RC4.

Storage Bucket (Public Cloud):



- Contains all the file parts of all the files.
- Each file part is identified using the unique ID.
- Also contains the key files of RC4 encrypted files.

Decrypted File:

