Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 06, June-2019

Data Privacy & Security in Premise Systems

An Innovative Approach

Utkarsh Deep
Student (batch of 2018), Computer Science & Engineering
Sir M Visvesvaraya Institute of Technology
Bengaluru, India.

Abstract—**This paper aims to provide a high-level blue-print of a data privacy & security design which is almost impossible to breach. This can be a game changer in designing password-based lock on data in premise systems.**

Keywords—Hashing, encryption-decryption, key, lock, random, data, recovery

I. INTRODUCTION

In this data driven 21th century, we are so concerned with security of non-premise data (data on a fixed server or cloud) that we forget to pay adequate attention in designing security in our applications for on-premise threats. While on-premise security is largely dependent on firewalls and anti-viruses, but software architects need to design data security for premise systems as well. There are many applications out there which hold critical data on premise. For Example - ERP Accounting Software (Like tally, marg, intuit etc.), IDEs which hold project code data on premise etc. This calls for designing a password-based data encryption-decryption design which should be theoretically impossible to breach unless you get to know the password by brute-force! How is this possible? A concise answer to this question is - This is possible only if you do not store password or its hash anywhere in the universe! Sounds exciting? Let's dive into the details.

II. IDEA – THE PROMISE

The first thing we should do before designing our security system is make a promise and stick to it. When we say we are introducing a security design which is theoretically impossible to breach, it means we are promising that – "*When you set a password lock on your data, the only way one can retrieve your data is by knowing your password*". There is absolutely no other way! Moreover, there is only one place where your password is stored – in your mind!

III. DESIGN BLUE PRINT – VISUALIZATION OF OUTCOME

Let us assume you have a large volume of data (called D). First, break this data into several small manageable chunks (called C). Therefore, we have:

D = C1 + C2 + C3 + …………... + Cn

Now, before we start explaining entire thing, we need to understand that there are few components which needs to be explained before we proceed further.

<u>Hashing Algorithm</u>:

We need to write our own hash algorithm to hash individual data chunks (C1, C2, etc.) and store them to get a new dataset called 'Hashed Data'. Let us call individual hashes as H1, H2 etc. and this new data as HD (abbreviation for Hashed Data). Now, we have:

HD = (H1C1) + (H2C2) + (H3C3) + ……………. + (HnCn)

<u>Key-Based Encryption-Decryption Algorithm</u>:

We need to write our own key-based encryption-decryption algorithm to encrypt hashed data (HD) based on user's password (key) as well as to decrypt data when a password is given. Note that the two terms, password and key are used interchangeably in this document.

<u>Random Number Generator</u>:

We need to write our own random number generator which generates a positive integer in the range [1, n], where n is total number of data chunks we have. Note that the seed of this random number generator can be current date and current time, or it can be anything which makes the generation of numbers truly random.

Now, there are 2 scenarios:

A. When user sets a password lock on his/her data
B. When user tries to retrieve his data.

A. *When User Sets A Password Lock On His/Her Data*

When user provides a password to lock his data, we should use our key-based encryption algorithm to encrypt the data without storing user's password (Key) anywhere.

<u>Step 1</u>: Build Hashed Data (HD)

HD = H1C1 + H2C2 + H3C3 + ……………. + HnCn

<u>Step 2</u>: Feed (1) Password and (2) Hashed Data as input to the encryption algorithm

Encryption Algorithm (INPUT: HD + Key) = Encrypted Data

This can be better visualized by looking at Figure -1 below:

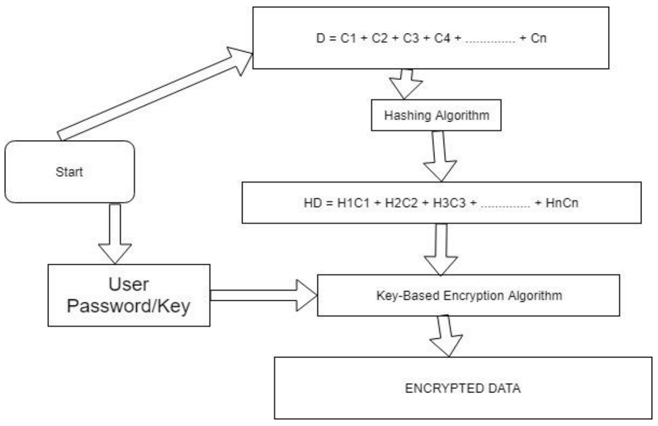
D = C1 + C2 + C3 + C4 + ... + Cn
Hashing Algorithm
Start
HD = H1C1 + H2C2 + H3C3 + ... + HnCn
User Password/Key
Key-Based Encryption Algorithm
ENCRYPTED DATA

Fig. 1. When User Sets Password on Data

B. *When User Tries To Retrieve His Data*

Since we are not storing user's password or its hash anywhere, it becomes extremely critical to understand the work-flow of validating correct password when user wants to retrieve his/her data. When user wants to retrieve data using password, following events take place:

Step 1: Decryption algorithm will decrypt the encrypted data using the password entered by user.

Decryption Algorithm (INPUT: Key + Encrypted Data) = HD

At this point, we are uncertain about the authenticity of Hashed Data (HD). This is because, we don't know whether user has entered correct password or not!

Step 2: We calculate the number of chunks (n) for HD and break the given HD into separate HCs like this:

HD = H1C1 + H2C2 + H3C3 + …………... + HnCn

Step 3: Call random number generator which will generate a random positive number in the range [1, n]. Suppose the output of random number generator is 'i'.

Step 4: For example, assume i = 7. Then we will pick 7th chunk – H7C7 and compute hash for C7. The computed hash should exactly match H7, otherwise password entered by user is said to be wrong and he will be asked to try again.

This can be better visualized by looking at figure-2.

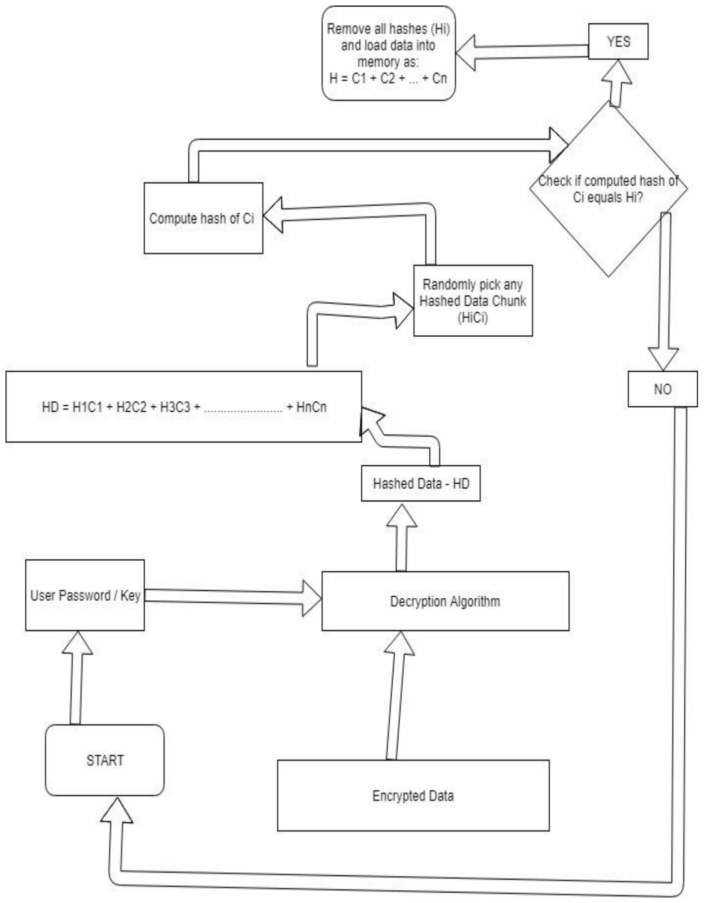
Remove all hashes (Hi) and load data into memory as: H = C1 + C2 + ... + Cn
YES
Check if computed hash of Ci equals Hi?
Compute hash of Ci
Randomly pick any Hashed Data Chunk (HiCi)
NO
HD = H1C1 + H2C2 + H3C3 + ... + HnCn
Hashed Data - HD
User Password / Key
Decryption Algorithm
START
Encrypted Data

Fig. 2. When User Tries to Retrieve His Data

a.

IV. ADVANTAGE & DISADVANTAGE

No password recovery mechanism:

Since user password or its hash is not stored anywhere, no one in this world can recover user's data in case he/she loses his/her password. This is both an advantage and disadvantage for the user.

(1) Advantage: This should give a sense of assurance to the user that no one can access his data because only he knows the correct password.

(2) Disadvantage: If user accidentally forgets his password, he will lose all his data.

V. ACKNOWLEDGEMENT

• Sincere thanks to my peers for their patience when I was explaining this approach to them.
• My parents – who have always been my core source of strength.
• My former Head of Dept. – Dilip Sen sir, who have always been my source of inspiration since I joined college.