AMERICAN JOURNAL OF AGRICULTURAL SCIENCE, ENGINEERING, AND TECHNOLOGY (AJASET)

ISSN: 2158-8104 (ONLINE), 2164-0920 (PRINT)

VOLUME 6 ISSUE 2 (2022)

Indexed in

PUBLISHED BY: E-PALLI, DELAWARE, USA

American Journal of Agricultural Science, Engineering, and Technology (AJASET)

Volume 6 Issue 2, Year 2022
ISSN: 2158-8104 (Online), 2164-0920 (Print)
DOI: https://doi.org/10.54536/ajaset.v6i2.368
https://journals.e-palli.com/home/index.php/ajaset

Mathematical Solution of ULTE Based Salt Generation Password Based Key Derivation Function (PBKDF)

Md. Alam Hossain[1*], Ahsan-Ul-Ambia[2], Nazmul Hossain[1], Mustafizur Rahman Akhond[1], Md. Nasim Adnan[1], Sayed Md. Galib[1]

Article Information

Received: July 02, 2022

Accepted: July 06, 2022

Published: July 07, 2022

Keywords

ULTE, PBKDF, PMMLCG, GSFRG, Super-prime, ULSI

ABSTRACT

In modern life, users use the cloud spaces for storing important data and valuable information. To avoid security problems, every cloud storage provider requires help from several security mechanisms. Every mechanism has its own security scheme. Most of them use the PBKDF (Password-Based Key Derivation Function) and Salt for generating encryption keys. By analyzing, we found that unique and random Salt plays a vital role for generating keys. But to create powerful keys, we must avoid pseudorandom generators for making salt because hackers can easily crack the key by using it. We have to take inputs from beyond expected sources. We proposed an algorithm by which we can generate Salt by using the client's real-time environmental information known as Location Information of user, User Time Information with Entropy Data at key generation time. In this paper, we focus on simulation result and compare each algorithm with the proposed algorithm ULTE. Simulation result represents the whole statistical distribution as stated in the algorithm. The result shows that ULTE has provided excellent performance than other algorithms.

INTRODUCTION

Data or information is growing day by day for different purposes. It is difficult to manage all the information by users using physical storage (Dillon et al., 2010 and Anitha Y, 2013). For this reason, Cloud plays a significant role in storing (Pring et al., 2009) and retrieving information. Users get extra space from the cloud by avoiding physical space (Numrmi et al., 2008). Cloud provides extra space by allowing Google Drive, Dropbox, One drive, Cloud Me etc. As the development of communication technologies are expanding day by day, resources of original information become not plentiful. It is necessary to acquire adequate knowledge on the System Requirement Specification (SRS), a structured collection of information that embodies the requirements of a system for implementing any communication tools (Hasan et al., 2021). But the communication specifications are not easy and not satisfactory. That is the reason of inability to communicate easily with others by different networks. Security issue is the most concerning factor in the current world (Sinha, N. and L. Khreisat, 2014) . Encryption key for security is a vital issue on communication. By analyzing we got that Random number generation process is more reliable on network environment. So, a new random number generation process ULTE (User Location Time and Entropy) is proposed (Alam Hossain et al., 2017, Md. Alam Hossain et al., 2017). It is based on entropy, time and user location data. This technology helps us to think positive about security and it is very convenient to our lives.

To protect the communication security encryption algorithm as well as encryption key is one of the most important factors (Somani et al., 2014). Generally key generation and random number generation are the two types of key generation steps. Random number is responsible for keys security. Low correlation and even distribution are the main characteristics of good random numbers. The drawback is that hacker or attacker can get it by anticipation or other means and then access the network, but it will not be an easy task for this method. In this method user location is frequently moving from one place to another and not only the location but also its corresponding time. Side by side the entropy data varies depending on system. So it is not possible to guess accurate number for hackers.

Nowadays, people mainly use cloud computing for application services and storing information. User name and passwords used for authentication and authorization by users. When user creates a new account with his/her name (user), a password at that time CSPs maintains its purpose very strongly. User Location Time and Entropy (ULTE) will give a high level security for user. For this reason it has a great acceptance than other procedures such as ULSI, PMMLCG, GSFRG, and Super-prime. Simulation results as well as comparison show the ULTE method has better impact than others.

To avoid various types of security threats and to ensure data security new types of applications are developed called Client Side Encryption Tools (CSETs). Every CSET has its own security schemes and essential features (Lin et al., 2009, Chen et al., 2011) and these tools are widely used by the users. Client Side Encryption tool uses encryption technique which transforms data files into a special form called cipher-text and then upload the files in the CSPs storage. The data files are being encrypted with encryption algorithms and encryption keys in encryption part. The securities of data files which are encrypted generally depends on the encryption algorithms and keys. Without decrypting the data files anyone can't understand cipher-text data files which is also called Decryption.

[1] Department of Computer Science & Engineering, Jashore University of Science & Technology, Bangladesh.
[2] Department of Computer Science & Engineering, Islamic University, Kushtia, Bangladesh.
[*] Corresponding author's e-mail: alam@just.edu.bd

Decryption is the opposite operation of encryption process. Decryption translate the cipher text to the plain text. This procedure depends on the decryption algorithms and decryption keys .CSETs is generally used for uploading the encrypted data files in cloud storage. CSET also downloads the data files from the cloud and decrypts the encrypted data files to the original form.

Through Password-Based Key Derivation Functions (PBKDF) Encryption and Decryption keys are propagated and maintained in Client Side Encryption Tools. In PBKDF every keys are generated manually. Since it is permeable making keys using random generator functions and by which hackers can easily crack keys. For increasing randomness and making each key more secured, generating from user's secret password with additional input. Each PBKDF is called by the choice of a fixed iteration count and Pseudorandom Function (PRF). A password, an unsecured random number called salt, and an indication of the targeted length of the key in bits which are included as input to an execution of PBKDF. Key cracking probability of user's password increases by guessing but are not completely cracked. Hackers have to spend a lot of time to crack key which is made from PBKDF. A key will be broken perfectly if Pseudorandom Function (PRF, password), number of iteration count salt and desired length of the key are accomplished. If one of the parameters is not accomplished then it is almost impossible to crack. Every each parameters of PBKDF are essential for making random and unique key. Except salt other parameters could be thought since salt is a random number input. So salt would be produced as true random number. Finally entropy sources such as HDD or thermal noise seek time make salt true random number.

Password-Based Key Derivation Functions (PBKDF)

It is necessary to encrypt various files at a time by a client for cloud computing. To encrypt different files, encryption tools of client side need to produce different unique encryption keys. To ensure randomness and uniqueness of encryption keys, keys are produced or generated depending on client's password with a special algorithm named PBKDF (Password Based Key Derivation Function) (Subrata Kumar Das et al., 2014).

For achieving access to a restricted resource or system a password or a passphrase is chosen by a user which is a secret string of characters. Almost all users usually use their name, pet name, date of birth, phone number, birth place, and others as passwords or passphrases that are easily permeable by social engineer. The passwords which user chooses have ailing weak statistical randomness properties and if these chosen passwords used as cryptographic keys directly then it is easy for hackers to crack or guess the cryptographic keys easily. For security in storage devices, there are some situations such as; the password or passphrase is the only secret information available to the cryptographic algorithm to protect the important and valuable data from malicious users or hackers.

Key Derivation Functions (KDFs) are deterministic algorithms that are used to perform cryptographic keying material from a user's password or passphrase. Password-Based Key Derivation Functions (PBKDF) is a well-known and broadly used algorithm for producing cryptographic keys by using KDFs. Each PBKDF (Barker et al., 2015) in the family which is defined by the selection of a Pseudorandom Function (PRF) and a stable iteration count, denoted as C. The input to perform of PBKDF includes a password, denoted as P, a salt, denoted as S, and an indication of the targeted length of the MK in bits, denoted as kLen. The kLen value will be at least 112 bits in length. A generic diagram of the PBKDF is given in (Figure.1), and symbolically.

mk = PBKDF(PRF, C) (P, S, kLen)

The main theme of a PBKDF is to slow dictionary or brute force attacks on the passwords by enhancing the time needed to examine each password. An attacker with a list of similarly passwords can calculate the PBKDF using the known iteration counter and the salt. Production of Salt by any pseudo random number generation algorithms form it easy for malicious users for cracking the encryption keys.

Spending a significant amount of computing time for each try, it would become harder to apply the dictionary or brute force attacks by an attacker. Randomly-generated passwords are stronger than user-chosen passwords of the same length. The length and its randomness properties

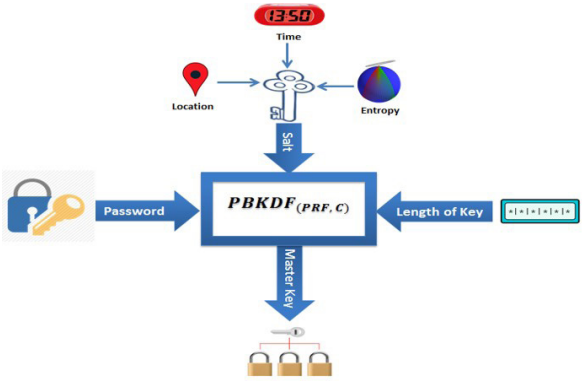

Figure 1: Generic diagram of the Password-Based Key Derivation Functions (PBKDF)

both are the strength of a password.

Ulte Algorithm

A. Environment Record Information for Salt

It is necessary for encrypting various files at a time by a client for cloud computing. To encrypt different files, encryption tools of client side need to produce encryption keys which are different and unique. To ensure randomness and uniqueness of encryption keys, produced or generated keys are depending on client's password with a special algorithm named PBKDF (Password Based Key Derivation Function). In the generation of encryption key process (random value) is used at each time in PBKDF (Chen, 2009). To generate Salt by any kind of generation algorithm of pseudo random number, it is easy for attackers or hackers or

malicious users for hacking encryption keys. In this paper, for generating Salt we subject an algorithm by using users actual environment information like User Location Information, (User) Time Information and Entropy Data of system that is calculated from HDD seek time or thermal noise or any kind of sources of noise at the generation time of key (Islam et al., 2012).

Location Information

For mobile end users, is not possible to predict user's location information. Moreover a user history of location that means location information cannot be anticipated or guessed through long time following or tracing the movement of users because it would be needed at the time of generation only. Geographical recorded environment information such as Latitude and Longitude. As mobile users can easily switch its location with respect to time and for making Salt, user's location information could be a great source. GPS antenna and services are included in almost modern mobile devices. Mobile devices position can be calculated in various ways like Wi-Fi network based, GSM-based, Hybrid mode, Handset-based, and Network-based in which two or more procedures are included (Nepal et al., 2016).

Time Information

With a short period of a given time if the user is stable or want to produce various keys at the corresponding session, its location of the information continues the same while the information of time is unstable or changeable. Each and every moment we got a unique time that cannot be revised. A time can't be same with another time by a year, by a month, by a day, by an hour, by a minute, by a second, even by a millisecond or microsecond or nanosecond.

Entropy Data: We can measure entropy data from HDD seek time or thermal noise of the users system. It is anticipated that this type of data cannot be guessed since attackers could not have the access of the system directly. It is a major source of bit strings (random). For Random Bit Generator (RBG) process and for the entropy source, root of the security is noise source. Mainly, it (noise source) supplies random bits in digital sample forms that are gained from a process of non-deterministic. Sample values which are gained from a noise source formed with bit strings having fixed length that can decide the entropy data of the output space.

B. Random Number Generation Process

User information of Location and Timing could be acquired by attacker or hacker by following and tracking continuously. In the Salt generation process it is needed to add the interface related information to ensure uniqueness and randomness of generated Salt. In these circumstances we need to add HDD seek time and thermal noise as entropy so that attacker cannot predict the Salt though in meantime location and time information. Generation process of Salt is figure out in the following for an individual salt. Full process will be periodic for multiple Salts in multiple times. For the purpose of calculation whole information will be shifted or translated into hexadecimal format before processing (Turan et al., 2016).

Location Information (LI) as history data

Users location information that includes latitude and longitude information in degrees-minutes-seconds format. For example: 243212, 1112334. This value or records indicates that user location latitude information is 24 degrees 32 minutes and 12 seconds. On the other hand longitude is 111 degrees 23 minutes and 34 seconds. These two information are combined written that means Latitudes and Longitudes information. So the above information will be 2432121112334 and converted hex value is 23645B7070E.

Time Information (TI) as history data

The recording time information in year-month-day-hour-minute-second format as shown in the following example. For example: 20161105224406. This record indicates 06 seconds, 44 minutes, 22 hours, on November 05, 2016. 19921105242424 indicate at 24 seconds, 24 minutes, 24 hours, on November 05, 1992. So the following hexadecimal of 19921105242424 is 121E3E66C938.

Entropy Data (ED) from system thermal noise or HDD

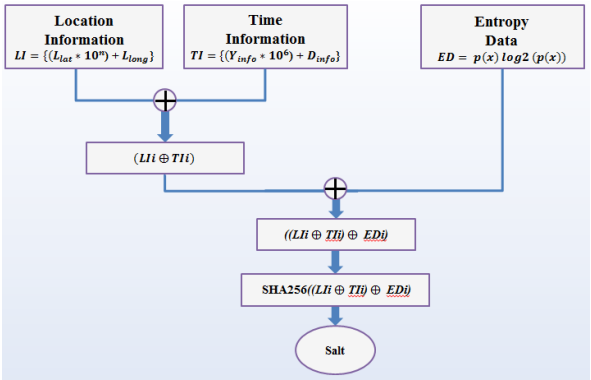

Figure 2: Salt Generation Process

seek time: Data from entropy is stored in 32 bit stream. For example 11010011000011110110011000111000 are recorded as Entropy data of thermal noise of system. Corresponding hexadecimal of 11010011000011110110 011000111000 is D30F6638. Then compute Hash value i.e. SHA256 ((LI ⊕ TI) ⊕ ED) to get 256 bits SALT.

Mathematical Representation

A. ULTE method

Location Information: User's location information (LI) is recorded as history data. This includes the latitude and longitude information. It calculates the latitude and longitude information as follows -

$L_{lat} = (D_1*10^4)+(M_1*10^2)+S_1$ ---------- (1)

$L_{long}=(D_2*10^4)+(M_2*10^2)+S_2$ ---------- (2)

Where L_lat be the latitude information (in decimal). D_1, M_1 and S_1 represents respectively the degrees, minutes and seconds for user's latitude information (in decimal).

Similarly, L_{long} be the longitude information (in decimal). D_2, M_2 and S_2 represents the degrees, minutes and seconds for user's longitude information (in decimal) correspondingly.

The equation for combined latitude and longitude information (LI) is-

$LI=\{(L_{lat}*10^n) + L_{long}\}$ ---------- (3)

Here, n be the numbers of longitude information in decimal format and will be like n=5, 6 or 7.

Time information: The time information (TI) of users includes the real time of the users in Year-Month-Day-Hour-Minute-Second format. The equation for measuring the yearly and daily information is-

$Y_{info}=\{(Y*10^4)+(M_{th}*10^2)+D\}$ ---------- (4)

$D_{info}=\{(H*10^4)+(M_{in}*10^2)+S\}$ ---------- (5)

Where, Y_{info} and D_{info} will be the yearly and daily information respectively. Y, M_{th} and D represents the present year, month and date respectively. D_{info} consists of H, M_{in} and S which represents hour, minutes and seconds correspondingly.

The resulting equation for TI will be-

$TI=\{(Y_{info}*10^6) + D_{info}\}$ ---------- (6)

Entropy: Consider a discrete random variable, the entropy of X is determined by computing the sum of

$p(x) \log2 (p(x))$ ---------- (7)

Where, x varies of all possible values for considering an observation of X and p(x) is the probability that an observation will have value x.

If N are the distinct possibilities for an observation of X, then the maximum possible value for the entropy of X is-

$\log2 (N)$ bits ---------- (8)

Min-entropy: Min-entropy of a discrete random variable X is a lower bound on its entropy and for a given finite probability distribution, p1……..pM, the precise establishment for the min-entropy m, is-

$m = - \log2 (\max (p1.....pM))$ ---------- (9)

Let xi be the digitized sample collected from the noise source which is represented like x1, x2, ..., xM be the outputs collected from the noise source. Let p(xi) be the probability that xi is provided at any given sampling time then we can get the output of the min-entropy is-

$- \log2 (\max p(xi))$ ---------- (10)

Algorithm: Algorithm for ULTE Salt Generation.

Step 1: [Initialize.] Read the current value of Location Information (LI) for both latitude and longitude in Degrees-Minutes-Seconds format.

Step 2: [Initialize.] Read current Time Information (TI) from the system in Year-Month-Day-Hour-Minute-Second format.

Step 3: [Evaluate.] Calculate (LI ⊕ TI).

Step 4: [Initialize.] Read the Entropy Data (ED).

Step 5: [Evaluate.] Calculate ((LI ⊕ TI) ⊕ ED).

Step 6: [Evaluate.] Calculate SHA256 ((LI ⊕ TI) ⊕ ED) to get 256 bits SALT.

Step 7: Exit.

B. Related Work (PMMLCG, GSFRG, Super-prime and ULSI method):

This section mainly focuses on GFSRG, PMMLCG, Super-prime, and ULSI method (Barker, E., and J. Kelsey, 2015).

i. GSFRG (H. Md. Alam et. al., 2012) (Generalized Feedback Shift Register) occurred register shift method. It also uses recursive formula that includes:

$X_{(i+p)}=X_{(i+q)} \oplus X_i$ (i=0,1,2….)

Here, p>q>0, Both p and q are integer. GSFRG have following steps:

$$\begin{cases} x_{i+1} = 3125 x_i \ mod(2^{35}-31) \\ r_i = \dfrac{x_i}{2^{35}-31} \end{cases} \quad (i = 0,1,2,3 \ldots\ldots)$$

$$\begin{cases} x_{i+1} = 16807 x_i \ mod(2^{31}-1) \\ r_i = \dfrac{x_i}{2^{31}-1} \end{cases} \quad (i = 0,1,2,3 \ldots\ldots)$$

Produce p random integers $(x_0,x_1,\ldots\ldots,x_{(p-1)})$, which are in (0,2p-1);

Then generate $x_{(i+p)}$ (i=0,1,2….);

Let, $r_i = x_i/2^p$ (i= 0, 1, 2……..), then $\{r_i\}$ is random sequence.

ii. Hutchinson present the PMMLCG (Lewis T. G., and Payne W. H., 1973) (Prime Modulus Multiplicative Linear Congruential Generator) methods which include:

iii. Super-prime method (Demirkol E., 2009) is a very special prime number. We enlist prime M, if m-1/m is not a complex proper fraction and the period is M-1, then M is a super prime. If Z belongs to $\{Z_i \mid Z_i \in N, 0<Z\_i< M\}$,choose initialization value $Z_i \in Z$:

$Z_{(i+1)}=10*Z_i \pmod M$ (i=1,2,3….)

Integer cycle sequence {Z_(j+s)} is the remainder, here $S \geq 0$, j=0, 1, 2, 3….. (M-1), let $r_i=Z_i/M$, then $\{r_i\}$ is considered as uniformly distributed pseudo-random sequence in the interval (0, 1).

iv. ULSI method (Demirkol E., 2009) uses environment record information for the purpose of generating random numbers. Its include spectrum information, geography and time information.

For N terminals information of location can be defined as:

$L=[L_1,L_2,\ldots\ldots\ldots L_N]$

Every terminals location is L_i, so $L_i= L_i0+ \Delta_{Li}$ (Here Δ_{Li} is random distributed)

Spectrum information is shown as

$B=[b_1,b_2\ldots\ldots..b_N]$

Spectrum information for the base station M is described as

$B_{bs}=[b_{bs1},b_{bs2},\ldots\ldots..b_{bsM}]$

Location record is constant because base station is stable or fixed. For the simulation process imagine that spectrum information B, terminal location information L obey the Gaussian distribution $N(\mu,\sigma^2)$.

Simulation

Simulating salt generation according to the introduced ULTE Salt Generation Algorithm is tabulated in this section. Simulation is accomplished using various random data of Time Information, User's Location Information, and Entropy Data. Result of every operation is

Table 1: User Location Information (Decimal, Hexadecimal)

Case	Location Information in Decimal	Location Information in Hexadecimal (LI)
1	1021506890655	3632A9574AB
2	3254171263209	B65D142D09
3	1212345885734	11A45722426
4	6526301542015	5EF85E2967F
5	2945361113927	2ADC533F347
6	1725202664903	191AE178DC7
7	3705191321321	35EAE9CCEE9
8	2754091394639	2813CA3A64F
9	1693750924126	18A5B6C135E
10	8259001873330	782F2DF73B2

Table 2: User Time Information (Decimal, Hexadecimal)

Case	Time Information in Decimal	Time Information in Hexadecimal (TI)
1	20170606052855	125855CF05F7
2	20170608111736	125855EE7078
3	20170615093549	12585658F92D
4	20170617020851	1258567661B3
5	20170618064616	125856864EE8
6	20170620105924	125856A574C4
7	20170622084403	125856C3A533
8	20170623073141	125856D2BB75
9	20170625062357	125856F115D5
10	20170629051238	1258572DF366

tabulated in the resembling table. Time Information and Environmental Information like location is recorded in decimal form and converted into hexadecimal for calculation purposes, also Entropy Data is recorded in binary form and converted into hexadecimal.

Recorded decimal value and equivalent hexadecimal value of user's Location Information are tabulated in (Table I). Location values are recorded when user's need to generate Cryptographic keys as well as Encryption and Decryption keys. Recorded decimal value and equal hexadecimal value of user's Time Information are tabulated in Recorded decimal value and equivalent hexadecimal values of user's Location Information are tabulated in (Table II).

Table 3: Entropy Data (Decimal, Hexadecimal)

Case	Entropy Data in Binary	Entropy Data in Hexadecimal (ED)
1	10001010101001101010101101010101	8AA6AB55
2	00101011000101101010011011010101	2B16A6D5
3	11100110101001011010101010010101	E6A5AA95
4	11010100101001101001001001001101	D4A6924D
5	10111010100110000010010101010101	BA982555
6	10110010100100101000011101010011	B2928753
7	11001011010101011010100001010101	CB55A855
8	10110010101001100101110101010101	B2A65D55
9	10110100111000101011010011010101	B4E2B4D5
10	11101010101011010100100100101010	EAAD492A

Entropy data is measured from system thermal noise signal. Thermal noise signal is captured in analog signal and then converted in digital form. Recorded binary

Table 4: Results of XOR operations of Location Information and Time Information

Case	LI (Location)	TI (Time)	(LIi ⊕ TIi)
1	3632A9574AB	125855CF05F7	113B7F5A715C
2	B65D142D09	125855EE7078	12EE08FA5D71
3	11A45722426	12585658F92D	1342132ADD0B
4	5EF85E2967F	1258567661B3	17B7D394F7CC
5	2ADC533F347	125856864EE8	10F593B5BDAF
6	191AE178DC7	125856A574C4	13C9F8B2F903
7	35EAE9CCEE9	125856C3A533	1106F85F6BDA
8	2813CA3A64F	125856D2BB75	10D96A711D3A
9	18A5B6C135E	125856F115D5	13D20D9D068B
10	782F2DF73B2	1258572DF366	15DAA5F280D4

Table 5: Results of XOR operation of Entropy Data with the output of XOR operations of Location Information and Time Information

Case	(LIi ⊕ TIi)	(ED)	((LIi ⊕ TIi) ⊕ EDi)
1	113B7F5A715C	8AA6AB55	113BF5FCDA09
2	12EE08FA5D71	2B16A6D5	12EE23ECFBA4
3	1342132ADD0B	E6A5AA95	1342F58F779E
4	17B7D394F7CC	D4A6924D	17B707326581
5	10F593B5BDAF	BA982555	10F5292D98FA
6	13C9F8B2F903	B2928753	13C94A207E50
7	1106F85F6BDA	CB55A855	1106330AC38F
8	10D96A711D3A	B2A65D55	10D9D8D7406F
9	13D20D9D068B	B4E2B4D5	13D2B97FB25E
10	15DAA5F280D4	EAAD492A	15DA4F5FC9FE

value and equal hexadecimal value of Entropy data which recorded from user's system thermal noise are tabulated in (Table III).

XOR operation is performed on Location Information (LI) and Time Information (TI) and results are tabulated in (Table IV). Location information and Time information are converted in hexadecimal for operation and outputs of XOR operation in hexadecimal.

XOR operation of Entropy Data with the output of XOR operations of Location Information and Time

Table 6: Results of SHA256 ((LIi ⊕ TIi) ⊕ EDi) operation

Case	((LIi ⊕ TIi) ⊕ EDi)	SHA256 ((LIi ⊕ TIi) ⊕ EDi)
1	113BF5FCDA09	5408C2E0B200094472AB2C1D27B421F1A04F832F503619F3B75BDE5B8A5B3CFB
2	12EE23ECFBA4	2706DCF878CFA0E8937C19D271DFD88E0F5E69194AC35770EC865CC6A5FD7FE3
3	1342F58F779E	BBF8ACC1BC188A986B0C498307915602F4B44D01B49E3C62C9D4738F7D05E289
4	17B707326581	7061659413C4031F746DE2E77EA89692675589D5A1E0EBED69053E22BFB4BF1C
5	10F5292D98FA	CDF53EAE6F198782D987CE39C139333D4511A3E5FB613F2BFB9A919D8CC81A07
6	13C94A207E50	53F287EFE3A67FC364EF103601BA54762F5719637E7D9D4944A2459A2A228DFB
7	1106330AC38F	49DF2F13204A92D9098FC8748203B135A59D436B0132F61A830ECD3A36B61EFE
8	10D9D8D7406F	1998086EEBA0808AF3BEDCE307D3D50BF985CE8D6677E18984E043FA55E26B09
9	13D2B97FB25E	1148B05EC75884F7559E7F5CB819A1C710F902E1D174843540316E60D3DE87E3
10	15DA4F5FC9FE	D0C42D2FB7A6C005EF37CCE4740DB2422ACEB8379E998B338DE4EE3917E68F85

Table 7: Results of SHA256 ((LIi ⊕ TIi) ⊕ EDi) operation

Case	SHA256 ((LIi ⊕ TIi) ⊕ EDi)	BINARY VALUE
1	5408C2E0B200094472AB2C1D2 7B421F1A04F832F503619F3B7 5BDE5B8A5B3CFB	1010100000010001100001011100000101100100000000000001001010001 0001110010101010110010110000011101001001111011010000100000111 1100011010000000100111110000011001011110101000000110110000110 0111110011101101110101101111011110010110111000101001011101100 11110011111011
2	2706DCF878CFA0E8937C19D2 71DFD88E0F5E69194AC35770 EC865CC6A5FD7FE3	1001110000011011011100111110000111100011001111101000001110100 01001001101111110000011001110100100111000111011111110110001000 1110000011110101111001101001000110010100101011000011010101110 1110000111011001000011001011100110001101010010111111101011111 1111100011
3	BBF8ACC1BC188A986B0C498 307915602F4B44D01B49E3C62 C9D4738F7D05E289	10111011111110001010110011000001101111100000110001000101010011 0000110101100001100010010011000001100000111100100010101011000 000010111101001011010001001101000000011011010010011110001111 0001100010110010011101010001110011100011110111110100000101111 10001010001001
4	7061659413C4031F746DE 2E77EA89692675589D5A1 E0EBED69053E22BFB4F1C	11100000110000101100101100101000000100111100010000000011000111 110111010001101101111000101110011101111110101010001001011010 010010011001110101010110001001110101011010000111100000111010 111110110101101001000001010011111000100010101111111011010010 11111100011100
5	CDF53EAE6F198782D987 CE39C139333D4511A3E5FB6 13F2BFB9A919D8CC81A07	1100110111110101001111101010111001101111000110011000011110000 010110110011000011111001110001110011100000010011100100110 01100 111101010001010001000110100011111001011111101101100001001111 110010101111111011100110101001000110011101100011001100100000 01101000000111
6	53F287EFE3A67FC364EF103 601BA54762F5719637E7D9D4 944A2459A2A228DFB	1010011111100101000011111110111111100011101001100111111110000 110110010011101111000100000011011000000001101110100101010001 110110001011110101011100011001011000110111111001111101100111 010100100101000100101000100100010110011010001010100010001010 00110111111011
7	49DF2F13204A92D9098FC87 48203B135A59D436B0132F61A 830ECD3A36B61EFE	1001001110111111001011110001001100100000010010101001001010110110 010000100110001111110010000111010010000010000000111011000100 110101101001011001110101000011011010110000000100110010111101 100001101010000011000011101100110100111010001101101011011000 01111011111110

8	1998086EEBA0808AF3BEDCE 307D3D50BF985CE8D6677E 18984E043FA55E26B09	1100110011000000010000110111011101011010000010000000010001010111100111011110110111001110001100000111110100111101010100001010111111100110000101110011101000110101100110011101111110000011000100110000100111000000100001111111010010101011110001001101100001001
9	1148B05EC75884F7559E7F5C B819A1C710F902E1D1748435 40316E60D3DE87E3	1000101001000101100000101111011000111010110001000010011110111010101011001111100111111101011100101110000001100110100001110001110001000011111001000000101110000111010001011101001000010000110101010000000011000101101110011000000110100111101111010000011111100011
10	D0C42D2FB7A6C005EF37CC E4740DB2422ACEB8379E99 8B338DE4EE3917E68F85	1101000011000100001011010010111110110111101001101100000000000101111011110011011111001100111001000111010000001101101100100100001000101010110011101011100000110111100111101001100100010110011100011011110010011101110000111001000101111110011010001111100000101

Information are performed and tabulated in Table 5.

In Table 6 the results are tabulated after calculating Hash of ((LIi⊕TIi)⊕EDi) using Secure Hash Algorithm –SHA256. Output of SHA256 is 256 bits and they converted in hexadecimal.

Results of Hash function SHA256 of ((LIi ⊕ TIi) ⊕ EDi) is converted into binary value and tabulated in the Table 7.

PARAMETER TEST AND RESULT

A. Methods

1. Random Key Generation: For preventing key generation that can be predictable by hackers, Generation of keys must have to random. However, keys are generated by computer software are never produced in

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

a truly random manner. At best, software-key generators usage pseudo-random processes to assured that virtually none can prophesy what keys are going to be produced. With our Cloud Salt Generation Process, we produce random keys for each key. Because every key generated from ULTE is different from each other's. To show the randomness between different keys, we use the standard omission method to simulate between different keys. The formula for Standard Deviation:

2. This test mainly covers mean test statistics, variance test and second moment test of the sequence of random number generator and examine if it exist great differences comparing with theoretical value.

i) Mean test: It examine the distribution of uniform random sequences in [0,1] interval and compare all sample of its mean value with the theoretical distribution of mean 1 / 2, for the identically free distributed random sequences, if the sample value is enough or sufficient then the theoretical value should be tends to 1 / 2.

ii) Variance test: It involves judging the uniformity of the statistical distribution and its theoretical variance should be tends to 1 / 12.

iii) Second moment test: Judging the random sequences

Simulation Result for Randomness between different number of keys

Number of Samples	Standard Deviation
2	9.21656E+11
3	8.56713E+11
5	6.80998E+12
7	3.42008E+12
10	2.91977E+12
6	1725202664903
7	3705191321321
8	2754091394639
9	1693750924126
10	8259001873330

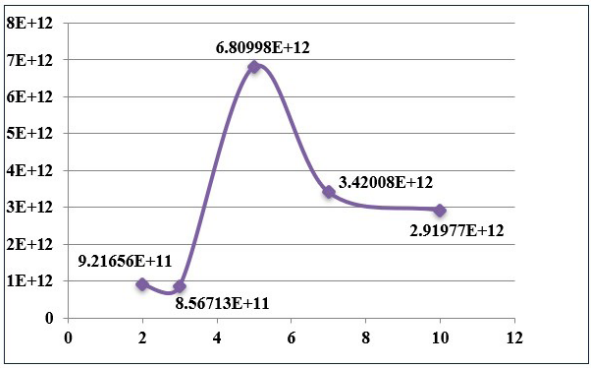

Figure 3: Simulation Result for Randomness between different number of keys

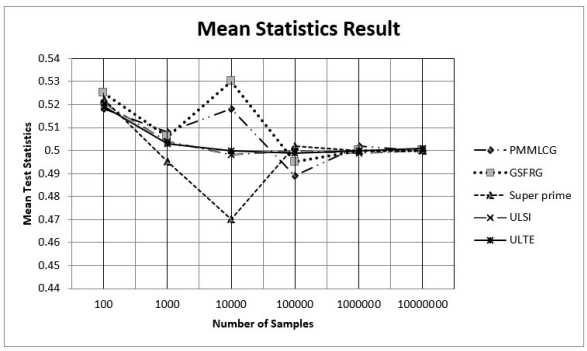

Figure 4: Distribution of Mean Test Simulation Result of second moment distribution. It can measure its statistical distribution whether it is even. For free / independent and identically random sequences theoretical value closes to 1 / 3.

B.Simulation Result

i) Mean Test Simulation Result

1.Simulation Result for Randomness between different numbers of keys:

From the above figure 4 we can conclude that:

a)As the sample number increases, the five methods are very close to theoretical value of 0.5. Among the five methods ULTE is very close to 0.5 for the same sample number. When the number of sample is 100000, ULTE has better performance than others. However, rate of convergence is basically same for different methods.

b) Statistical distortion creates when number of sample is insufficient and as a result volatile part appears. All methods close to theoretical value when number of samples increases.

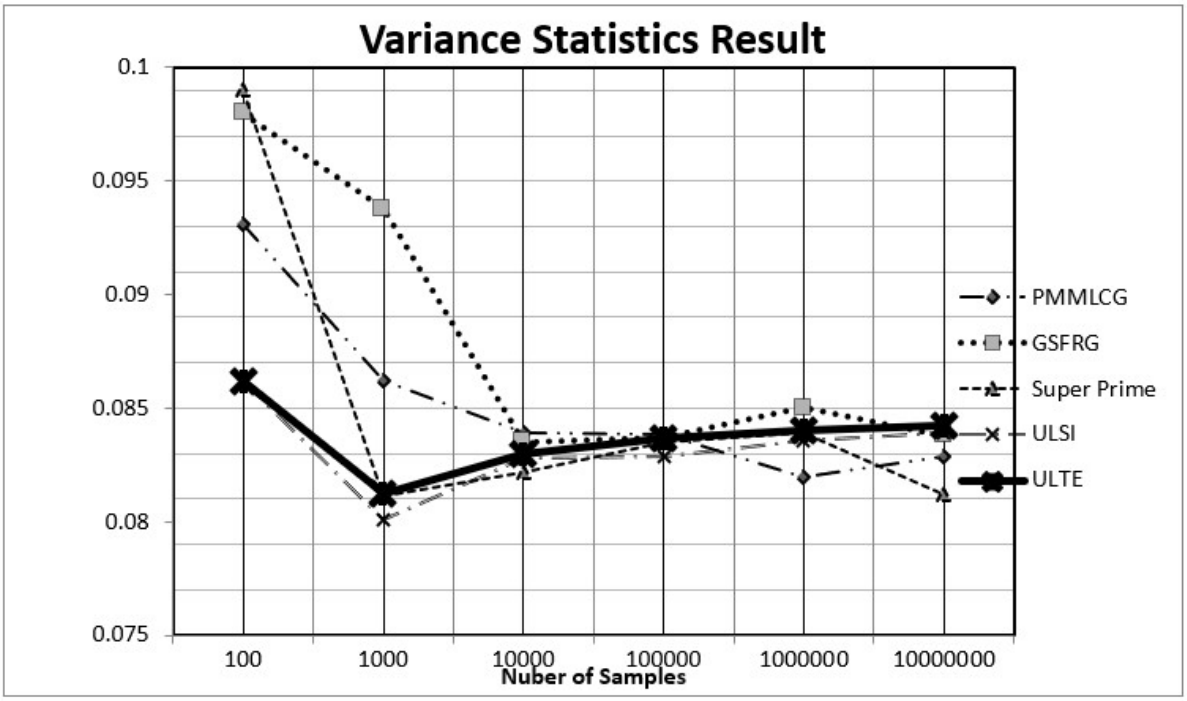

Figure 5: Distribution of Variance Test Simulation Result

ii) Variance test Result

From the above figure 5 we can say the following comparison:

a) When the number of sample value increases, every method is tend to the theoretical value 1 /12.

b) For the similar sample number, value of ULTE is closer to the theoretical than others. So we can say that ULTE has great performance than other methods which is signified.

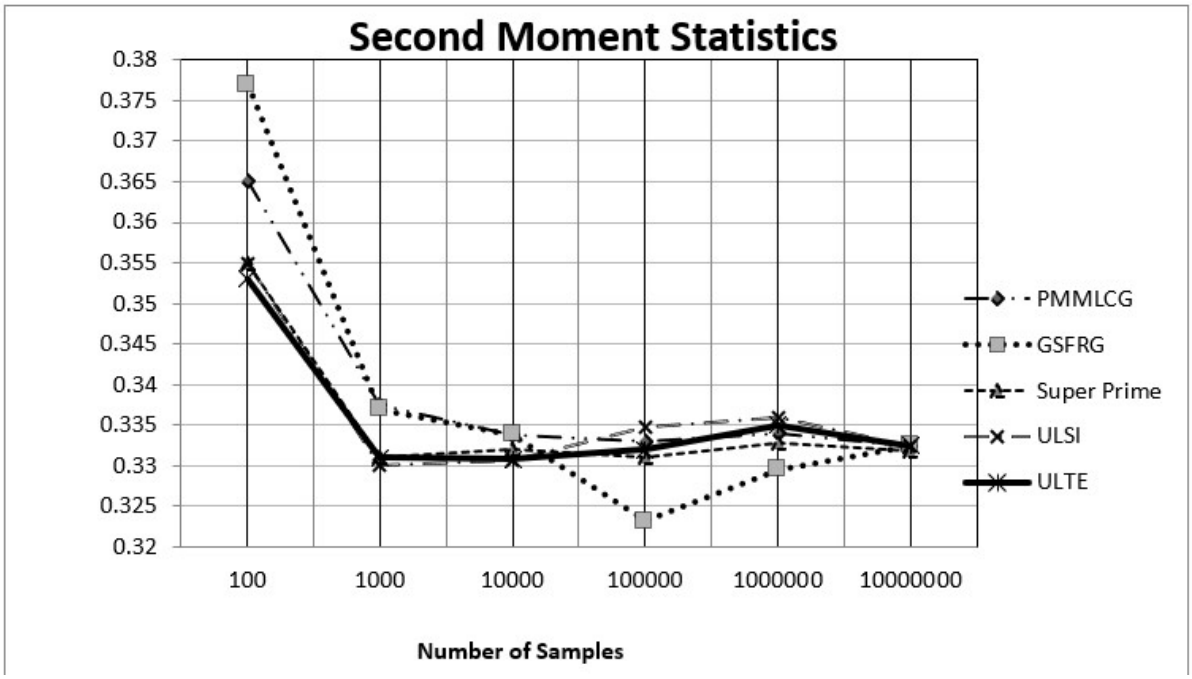

Figure 6: Distribution of Second Moment Test Simulation Result

c) Convergence of ULTE is comparatively faster than other methods which show the generation of random numbers by ULTE has greater smoothness.

C. Overview

From the previous figure, we can conclude that-

i. Each key generated from ULTE is different from each other's.

ii. ULTE will provide enough randomness among the keys.

iii. As a result users will get enough security for each of his data.

iv. ULTE has a better performance than others method

CONCLUSION AND FUTURE WORK

We design ULTE method by salt generation process and it is based upon location information of user, time information of user and system entropy. For mobile user, they frequently change their position from one place to another that means position of a mobile user is not stable and it is the main reason for hackers or attackers why they are not able to track the position of a user easily (Abdullah Al-Mamun et al., 2017). Performing the simulation result from the above figures we can draw a conclusion that ULTE has a better performance than others method (Aleem A. and C. R. Sprott, 2013, Das et al., 2014).

In future we will design an architecture of a client side encryption tool for heterogeneous cloud providers (different cloud providers use different APIs, synchronization method, etc.) (Chen D. and H. Zhao 2012, Zissis D. and D. Lekkas, 2012). In the terms of sharing the data with other users, the cloud storage providers are not fully trusted. So the encryption keys are managed in a decentralized environment and revocation is necessary when somebody leave from the share either willingly or forcibly (Wang et al., 2010, Pearson, 2009).

REFERENCE

Abdullah Al- Mamun, Shawon S. M. Rahman, Tanvir Ahmed Shaon and Md. Alam Hossain (2017). Security Analysis of AES and Enhancing its Security by Modifying S-Box with an Additional Byte. *International Journal of Computer Networks & Communications (IJCNC), 9(2). 69-88,* DOI: 10.5121/ijcnc.2017.9206.

Aleem A. and C. R. Sprott (2013). Let Me in the Cloud: Analysis of the Benet and Risk Assessment of Cloud Platform. *Journal of Financial Crime, 20(1), 6-24*. http://dx.doi.org/10.1108/13590791311287337

Anitha Y. (2013). Security Issues in Cloud Computing-A Review. *International Journal of Thesis Projects and Dissertations (IJTPD), 13, 1-6*.

Barker, E., J. Kelsey (2015). Recommendation for Random Number Generation Using Deterministic Random Bit Generators- Revision 1. Computer Security Division, *Information Technology Laboratory,* NIST SP 800-90A. http://dx.doi.org/10.6028/NIST.SP.800-90Ar1

Barker, E., J. Kelsey (2015). Recommendation for Random Number Generation Using Deterministic Random Bit Generators- Revision 1. Computer Security Division, Information Technology Laboratory, NIST SP 800-90A. http://dx.doi.org/10.6028/NIST.SP.800-90Ar1

Chen, S., S. Nepal, and R. Liu (2011). Secure Connectivity for Intra-cloud and Inter-cloud Communication". 40th *International Conference on Parallel Processing Workshops (ICPPW)* (pp. 154-159).

Chen L. (2009). Recommendation for Key Derivation Using Pseudorandom Functions. Computer Security Division, Information Technology Laboratory, NIST SP (pp. 800-108). http://csrc.nist.gov/publications/nistpubs/800-108/sp800-108.pdf

Chen D. and H. Zhao (2012). Data Security and Privacy Protection Issues in Cloud Computing. *The International Conference on Computer Science and Electronics Engineering, Hangzhou (pp. 647-651).*

Das, S. K., M. A. Hossain, M. A. Sardar, R. K. Biswas, and P. D. Nath (2014). Performance Analysis of Client Side Encryption Tools. *International Journal of Advanced Computer Research, 4(3),* 888-897. http://accentsjournals.org/PaperDirectory/Journal/IJACR/2014/9/20.pdf

Demirkol E., Mehta S., and Uzsoy R. (2009). Benchmarks for shop scheduling problems. *European Journal of Operational Research, 13, 137-141*. http://dx.doi.org/10.1016/S0377-2217(97)00019-2

Dillon, T., C. Wu and E. Chang (2010). Cloud Computing: Issues and Challenges. Proceedings of the 24th IEEE *International Conference on Advanced Information Networking and Applications (pp. 27-33).* Perth, WA.

H. Md. Alam, Ahsan-Ul-Ambia, Al-Amin and K. Rahamatullah (2017). Measuring Interpretation and Evaluation of Client Side Encryption Tools in Cloud Computing. Security, Privacy and Reliability in Computer Communications and Networks (pp. 207-239). River Publishers Series in Communications.

H. Md. Alam, Ahsan-Ul-Ambia, Md. Al-Amin, and H. Nazmul (2017). User Location Time and Entropy (ULTE) based Salt generation for Password Based Key Derivation Function (PBKDF) in Cloud Computing. *International Journal of Scientific & Engineering Research, 8(2),* 1399-1408.

H. Md. Alam, Md. Kamrul Islam, Subrata Kumar Das and Md. Asif Nashiry (2012). Cryptanalyzing Of Message Digest Algorithms Md4 And Md5 *International Journal on Cryptography and Information Security(IJCIS), 2(1),* 1-13.

Hasan, R., Prapti, D. R., G M, M. ., & Chakraborty, T. R. (2021). System Requirement Specification of Mobile Apps for shrimp farming in Shyamnagar of Bangladesh. *American Journal of Agricultural Science, Engineering, and Technology, 5(2),* 1–10. https://doi.org/10.54536/ajaset.v5i2.66

Islam, M. K., M. A. Hossain and M. A. Nashiry (2012). Security of cryptographicalgorithm sha and md5, *Institutional Engineering and Technology (IET), 2(2),* 28-33.

K. D. Subrata, H. Md. Alam, S. Md. Arifuzzaman, B. Ramen Kumar, and N. Prolath Dev (2014).

Performance Analysis of Client Side Encryption Tools. *International Journal of Advanced Computer Research (IJACR), 4(3)*, 888- 897. DOI: 10.19101/IJACR

Lewis T. G., and Payne W. H. (1973). Generalized Feedback Shift Register Pseudorandom Number Algorithm." *Journal of the ACM (JACM), 20(3)*, 456-468. http://dx.doi.org/10.1145/321765.321777

Lin, H. C., S. Babu, J. S. Chase, and S. S. Parekh (2009). Automated Control in Cloud Computing: Opportunities and Challenges. *1st Workshop on Automated control for data centres and clouds (pp. 13-18)*. New York, NY, USA.

Nepal, S., C. Friedrich, C. Wise, R. O. Sinnott, J. J. Jaccard, and S. Chen (2016). Key Management Service: Enabling Secure Sharing And Deleting Of Documents On Public Clouds. *Services Transactions on Cloud Computing (STCC), 4(2),* 15-31.

Nurmi, D., R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff and D. Zagorodnov (2008). The eucalyptus open-source cloud computing system. *12th International Conference on Cloud Computing and Its Applications (pp. 1-8)*. Washington, DC, USA.

Pring, B., R. H. Brown, A. Frank, S. Hayward and L. Leong (2009). Forecast: Sizing the cloud; understanding the opportunities in cloud services, Gartner Inc., Tech. Rep.G00166525. 27(3).

Pearson, S. (2009). Taking account of privacy when designing cloud computing services. The CLOUD '09 Proceedings of ICSE Workshop on Software *Engineering Challenges of Cloud Computing (pp. 44-52)*. IEEE Computer Society, Washington, DC, USA.

Sinha, N., and L. Khreisat (2014). Cloud computing security, data, and performance issues. *23rd Wireless and Optical Communication Conference (WOCC) (pp 1-6),* Newark, NJ.

Somani, U., K. Lakhani, and M. Mundra (2014). Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing. *1st International Conference on Parallel, Distributed and Grid Computing (PDGC - 2010)*, Delhi, India.

Turan, M. S., E. Barker, J. Kelsey, K. A. McKay, M. L. Baish, and M. Boyle (Jan. 2016). Recommendation for the Entropy Sources Used for Random Bit Generation. Computer Security Division, Information Technology Laboratory, NIST DRAFT Special Publication 800-90B. http://dx.doi.org/10.6028/NIST.SP.XXX

Wang, C., Q. Wang, K. Ren and W. Lou (2010). Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing. The IEEE 29th Conference on Computer Communications in INFOCOM, San Diego (pp. 1-9).

Zissis D. and D. Lekkas (2012). Addressing Cloud Computing Security Issues. *Future Generation Computer Systems, 28(3), 583-592*. https://doi.org/10.1016/j.future.2010.12.006