Bifid-Crypt

Secure storage and data-flow in cloud computing and IoT.

Open this website

Cyber Security J Component -- Implementation of bifid cipher in cloud computing

Research Paper

MATLAB-Based Software Tool for Implementation of Bifid Ciphers

Abstract

Cloud Storage is a service where data is remotely maintained, managed, and backed up. The service allows the users to store files online, so that they can access them from any location via the Internet. In recent days, almost everyone is moving towards cloud storage and cloud computing, According to a recent survey conducted with more than 800 business decision makers and users worldwide, the number of organizations gaining competitive advantage through high cloud adoption has almost doubled in the last few years and by 2017, the public cloud services market is predicted to exceed $244 billion. People in their daily life use cloud storage for saving their private data. Hence, need for the data to be secured is much more than the same in case of local storage. There are concerns with the safety and privacy of important data stored remotely. Any compromise in data will automatically destroy the privacy, which is the fundamental right for human beings.

Why BIFID?

Bifid is a Lightweight Encryption Algorithm for Secure data transfer and storage. The shortcomings of the cloud and IoT in terms of constrained devices are highlighted. There in fact exist some lightweight cryptography algorithms that does not always exploit security-efficiency trade-offs Therefore, we chose our topic to be security in storage and data- flow in cloud storage and cloud computing using this Algorithm. We hope our efforts in this project will be benefitial to society and public who have completely shifted their usage/storage/buisness to cloud computing.

Objective

Cloud Storage is a service where data is remotely maintained, managed, and backed up. The service allows the users to store files online, so that they can access them from any location via the Internet. In recent days, almost everyone is moving towards cloud storage and cloud computing, According to a recent survey conducted with more than 800 business decision makers and users worldwide, the number of organizations gaining competitive advantage through high cloud adoption has almost doubled in the last few years and by 2017, the public cloud services market is predicted to exceed $244 billion.
People in their daily life use cloud storage for saving their private data. Hence, need for the data to be secured is much more than the same in case of local storage. There are concerns with the safety and privacy of important data stored remotely. Any compromise in data will automatically destroy the privacy, which is the fundamental right for human beings.
Invented by Felix Blessed Virgin Delastelle (1840 - 1902), though divided microorganism word [Ame05 Kah67] has never been used for any "serious applications, it's become among the foremost widespread passwords "Amateur" cryptographers.

Proceedure

For Hash Function

This document shows the dataflow in our project

signup

Form the username and password

Hashing of password and then storing it to DB

password -> hash then store to DB hashing key --> will be private to the server

```
sample key =
| X A P U F |
| N C G Y K |
| I Z E O M |
| B L W V R |
| D H Q T S |
```

NOTE*

> output of hash function will be of 56 letters

length of password = x letters

concatString (n-letters) n=56-(x+1)

concatString: i = [0, n-1]

```
concatString[i] = addStr [ (floor((((i*2 + x)^3)/2)%16 ];
password.concat(concat-string);
```

> apply bifid cipher now to find the encrypted hash

```
addStr = {
        0: m,
        1: f,
        2: g,
        3: e,
        4: r,
        5: y,
        6: z,
        7: q,
        8: r,
        9: g,
        10: i,
        11: o,
        12: a,
        13: q,
        14: e,
        15: s
};
```

EXAMPLE

```
password: "Madhav" + 50 random letters
password[7] = '$'
required letters = [8 to 55]
n = 56 - (6+1) = 49 letters
concatString = [0 to 48]

concatString[0] = addStr[(floor((((0*4)^2)/3)))%16] = addStr[0] = m
concatString[1] = addStr[(floor((((1*4)^2)/3)))%16] = addStr[5] = y
....
....
....
password.concat(concatString)
```

Now the password will be of 56 letters

> Now apply bifid to find the encrypted hash of the given password
>
> > Store the password to the DB

Procedure for Bifid Cipher

> Private key will be the input from user. Polybius square is a 5x5 matrix which is used as the private key.

- Input the private key
- Put it into Polybius square letter by letter ignoring the repetitions.
- Fill the remaining english alphabets in the blank spaces of Polybius square (the letters I and the J share their position in the Polybius square)

```
  1 2 3 4 5
1 B G W K Z
2 Q P N D S
3 I O A X E
4 F C L U M
5 T H Y V R
```

- The message is converted to its coordinates in the usual manner, but they are written vertically beneath:

Example

```
F L E E A T O N C E
4 4 3 3 3 5 3 2 4 3
1 3 5 5 3 1 2 3 2 5
```

- They are then read out in rows (lower row is concatenated in the first row)

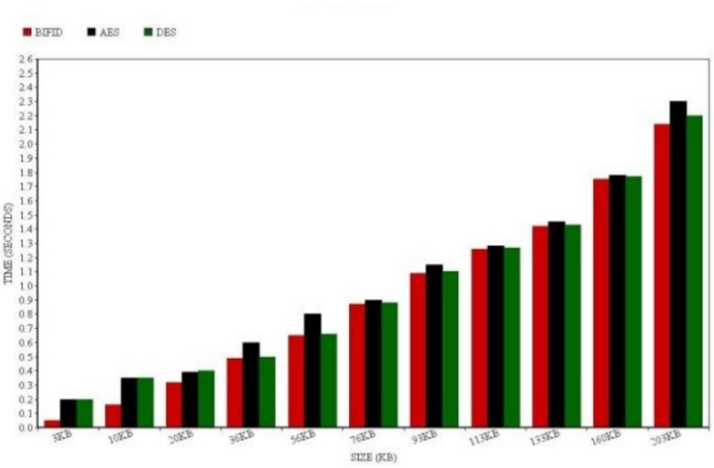
```
4 4 3 3 3 5 3 2 4 3 1 3 5 5 3 1 2 3 2 5
```

- Then divided up into pairs again, and the pairs turned back into letters using the Polybius square

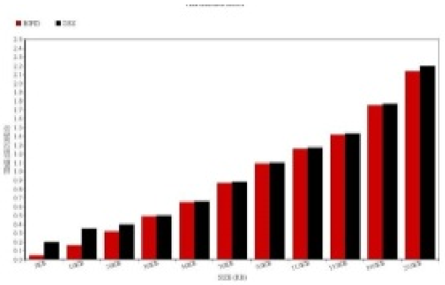
```
44 33 35 32 43 13 55 31 23 25
U  A  E  O  L  W  R  I  N  S
```

Time Phase Analysis

Comparison between BIFID, AES, and DES



Comparison between DES and BIFID



Comparison between AES and BIFID