https://dx.doi.org/10.4314/dujopas.v8i1b.3
ISSN (Print): 2476-8316
ISSN (Online): 2635-3490
Dutse Journal of Pure and Applied Sciences (DUJOPAS), *Vol. 8 No. 1b March 2022*

Database Security Framework Design Using Tokenization

[1]Rihanat Bola Agboola, [2]Zaharaddeen Salele Iro,
[3]Jamilu Awwalu, [4]Ibrahim Nasir Said

[1]ICT Directorate,
Federal University Dutse

[2,3]Department of computer science,
Faculty of Computing,
Federal University Dutse.

[4]Department of Electrical Engineering,
Newcastle Univeristy,
UK.
Email: naseeribro@gmail.com

Abstract
One of the challenging tasks in database management system today is protecting sensitive data that are complex in the database system. Most organization opt for implementation of data encryption which is one of the most effective security control mechanism, but data encryption only render information unreadable to those that do not have the secret decryption key. The core issue with data encryption techniques is that, it is reversibly. In these regards, sensitive data need to be tokenized first before encrypting for better security. Therefore, this work is introducing tokenization technique. Tokenization means replacement or substitution of sensitive data with a token, as an additional technique on database system for protecting sensitive data in all higher institutions and other firms that deal with sensitive data. A database security framework using tokenization was design, implemented using NetBeans 8.1 IDE, Java Platform SE binary (jdk-8u91-windows-x64) and xampp-windows-x64-7.3.7-0-VC15 (Apache and MySQL), an official IDE for Java 8 and then encrypt the tokenized data using AES before storing it in server 127.0.0.1 a localhost phpMyAdmin Database tools and evaluated with other security systems (encoding, hashing and encryption) on student database. Results shows that tokenization technique has 95% capacity to protect sensitive data compare to other security systems.

Keywords: Database security, Data encryption, decryption, Hashing, Tokenization.

INTRODUCTION
In today's world, data is generated at a very fast rate and the end point of such data is database. All the operations of data such as create, edit, delete etc. and maintenance are done using Database Management System. Considering the importance of data in organizations, it is essential to make sure that the data in the database is not vulnerable to hackers. According to the 2012 data breach investigations report published by Verizon Business, 96% of records breached in 2011 were taken from database servers. 56% exploited default or guessable identification while 40% used stolen login credentials. Also, according to another study among data professionals conducted by Unisphere Research in 2019, more than one-third of the 430 respondents install 37% critical patch updates within three months of release

*Author for Correspondence

(webhostingreeks.com). Most organization opted for implementation of only data encryption which is one of the most effective security control available as far as Database security is concern if properly implemented, but data encryption only render information unreadable to those that do not have the secret decryption key and the core issue with data encryption techniques is that, it is reversible (Kristen, 2017). Also some organizations try additional layer on their endpoint, all these may not work as expected or stop hacker from attacking the database, because only the strength of algorithm and computational power available to the hacker will determine how easily the data can be hacked.

An earlier research (Satyam, 2019) as part of overall security assessment designed a guideline for user to identify security flaws at an early stage before development. This work only took care of data encryption or hashing of sensitive data which is not enough as far as securing sensitive data is concern. Roger *et al*., (2021) shows how tokenization is presently in use to digitalize transaction security system, and also illustrate how other firms can apply tokenization to innovate their business. Babich, (2020) applied digital claim tokens in supply chain management for production, inventory, and financial controlling that enables a facilitated sharing, trading, and exchange among multiple stakeholders. Besides widely discussed implications for innovative information systems, the concept of tokenization has emerged to enable a system for management of assets ownership of peculiar digital representation (Hrga et *al 2020*). Shtybel, (2019) explores the benefits of tokenized private securities for improved issuance, trading, and settlement. As opined by Roger *et al*., (2021) existing research on the applicability of tokenization mainly concentrate on general potentials and challenges with strong emphasis on the financial sector such as banks, online transactions etc., That is because as far as database security is concern tokenization stands at the top in protecting sensitive data.

Over the years' cryptograph technics has been one way to make sure that confidentiality, integrity and availability data is secured and maintained using hashing, encoding and encryption. It is a concept that mainly keeps vital information that is subject to possible data breach safe and confidential but due to its reversibility, the process is not enough to secure data from hackers and that is why this research work is introducing tokenization process to higher institution system which could be used to secure their student database, clinic record and other sensitive data in the environment.

TOKENIZATION

Is replacement or substitution of sensitive data with a token, a highly secure method of protecting payment with a onetime number known as token that has no value or meaning to the account, because token is generated through random algorithms, which cannot be reversed or linked back to original sensitive data, it makes it difficult for hackers to place a value on the information they are looking for because there is no way they can decrypt or decode a token (flumer, 2020). Tokenization can be handle either by in house or a service provider but according to some comparative reports, the operation costs of in-house payment tokenization outweigh the overall cost of having a service provider (Linda, 2019).

Token Generation

Each time an information is sent to the token server, three steps are carried out. First, a token is generated. Second, the token and its corresponding ID that will be used to map the original data are stored together in the token database while the original data is completely removed from the environment and sent to the service provider. Lastly the token is reverted to the calling application (Securosis, 2018).

Token Storage

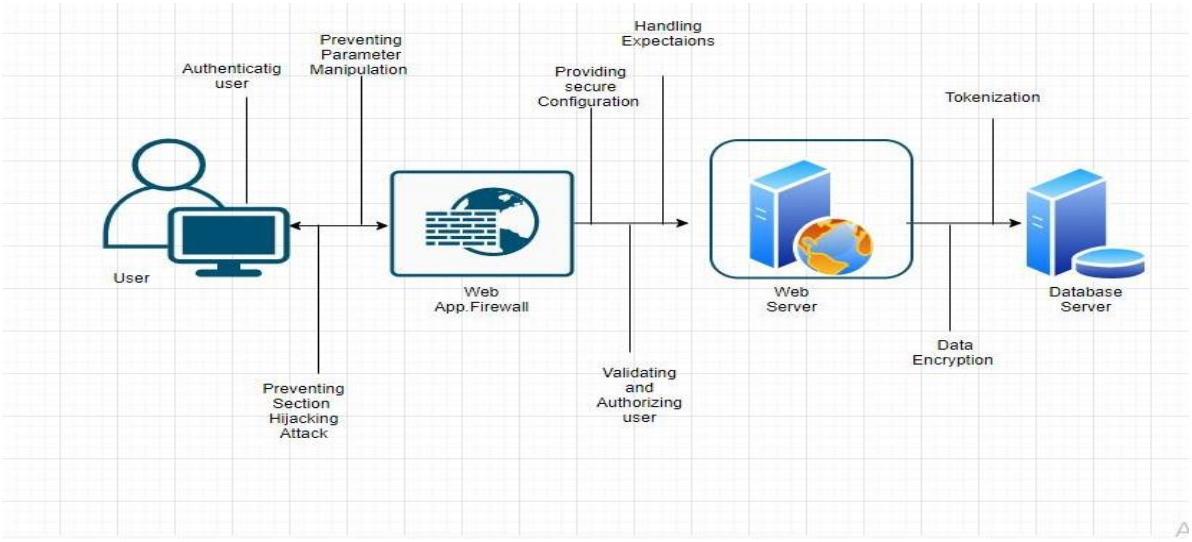
Token should to be flexible to handle several formats for the sensitive data they accept, such as personally identifiable information, student registration number etc. From previous research, Tokens along with the original data they represent, are stored within secured databases with limited access. Ideally the application should never store the original value and the token in the same environment, original data is better with the service provider (Securosis, 2018).

PROPOSED FRAMEWORK

System Architecture

The proposed framework architectural design using tokenization process in Fig.1, shows how data in a database or data vault could be secured.

Figure 1: Database Security Framework Design



The framework consists of four modules that are significant towards achieving an effective database security system which are Browser, Web Firewall Application, Web Server and Database.

Browser: The Browser also known as web browser is an application software that allows direct interaction between the user and the internet such as find, display, access, view websites, through Google Chrome, Mozilla Firefox, Microsoft Internet Explorer (ISG Tech, 2020).

The user will login using, user name and password into the internet this process is called Authentication Process.

Web Application Firewall: is a security firewall that is designed to allow authorize low risk that might not harm the network and if any harmful traffic is detected either from virus or a hacker trying to gain entry, fireworks blocks it immediately (GBHackers, 2019). When the user login using user name and password, its web application firewall that will check to prevent any parameter manipulation and section hijacking before authorization.

Web Server: A Web Server is a software and hardware devices that uses HTTP Hypertext (Transfer Protocol and also support), SMTP (Simple Mail Transfer Protocol) and FTP (File Transfer Protocol) to respond to client request made over the WWW (World Wide Web) by

displaying website content through storing, processing and delivering webpages to users. (Alexander, 2020).

Database Server: are highly powered computers that store and manage data for network users and devices**.** The database server holds the Database Management System (DBMS) and the databases. Upon requests from the client machines, it searches the database for selected records and passes them back over the network (Sam, 2021). Before data will be store in the database server, the original data passes through two processes which are Tokenization and Data Encryption Process.

Tokenization processes

1. The data that needs to be tokenize is passed by the application to the tokenization system along with authentication information.
2. The tokenization system validates the authentication information and if authentication fails, then the process stops and information (feedback) is sent to the deny access log error, this will let administrators to identify issues and manage immediately. But if authentication is valid, then the system proceeds to the next stage which is tokenization system as shown in Fig. 2.

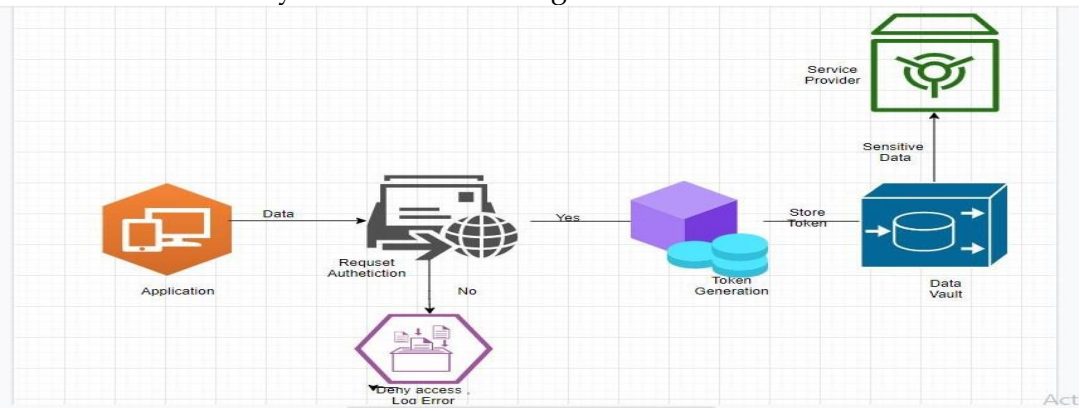


Figure 2: Tokenization Process

The tokenization system also generates token known as one-time password (OTP) base on one way cryptographic algorithms which are stored in highly secured data vault and the new token is passed through the application for data retrieval, while the sensitive data (Original Data) is highly protected with service provider, therefore, even if any hacker finds their way into the database and gets token it is as good as they have got nothing. As it is shown in Fig. 2 for a user to access any data in the database a token will be sent to the client to confirm the authentication of that user before granting an access. Porter's Algorithm is one of the most prominent tokenization techniques but suffers from accuracies during identification and efficiency. While in this work token identification is completely JAVA based.

Proposed Algorithm for Tokenization Process

Input (Di)
Output (Tokens)
Begin
Step 1:
Collect Input data to be tokenized (Di) where i=1, 2, 3…. n;
Step2:
For input request authentication Di;
// apply extract word process for all data i=1, 2, 3…n //
IF request authentication is YES THEN

Step 3:
GO TO Token generation
Step 4:
store token generated in data vault
Step 5:
original data will be passed to service provider system.
ELSE
Step 6:
Deny access log error
Step 7:
End

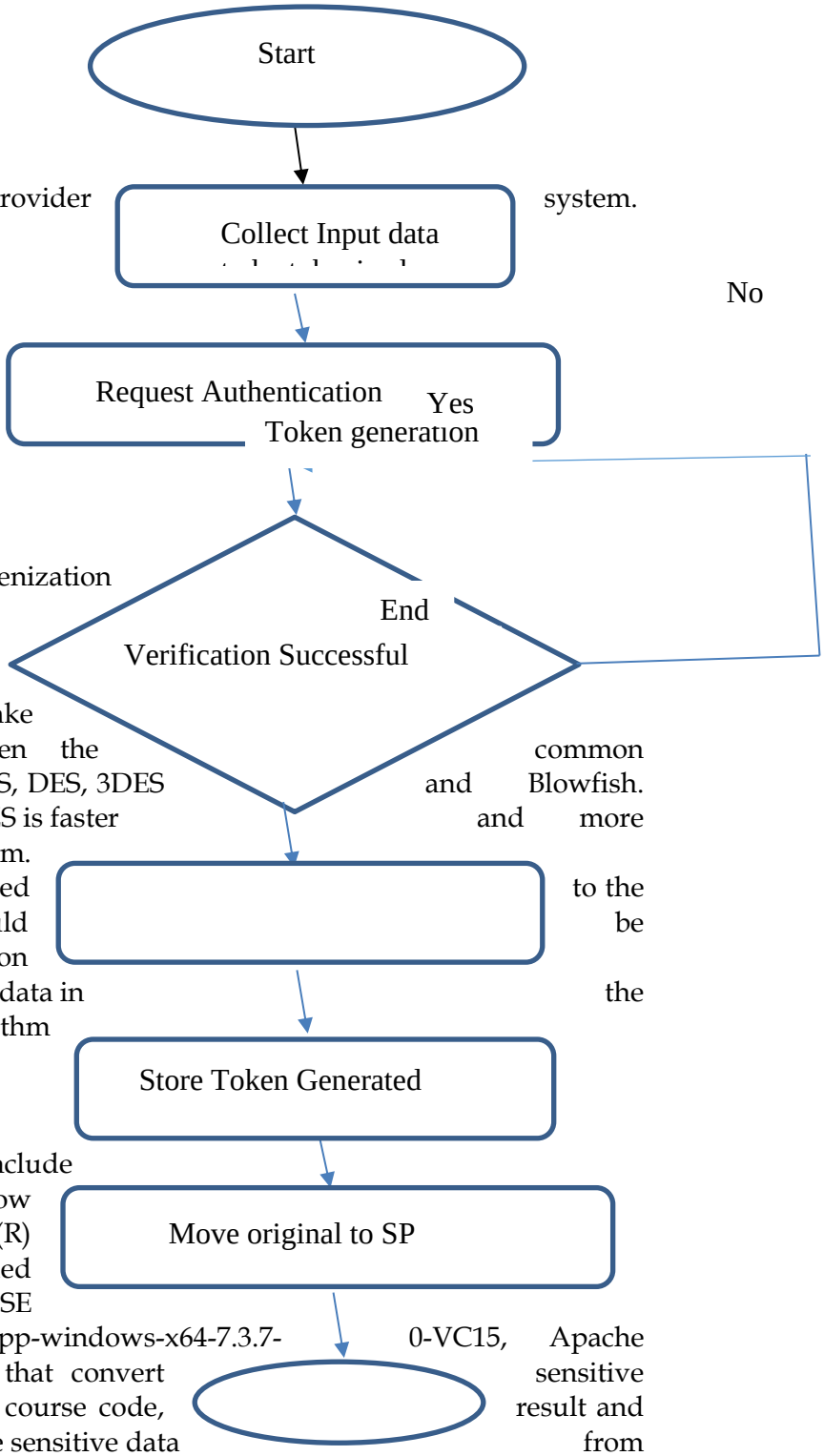


Figure. 3: Flowchart of Tokenization Algorithm

Data Encryption

By surveying a number of studies that make comparison in performance between the common encryption algorithm which include AES, DES, 3DES and Blowfish. The results of comparison stated that AES is faster and more efficient than other encryption mechanism. Therefore, the fact that token is not related to the original data does not mean that it should be exposed Immediately after tokenization process, both token generated and other data in the database are encrypted using AES algorithm (Kiramatullah, 2019).

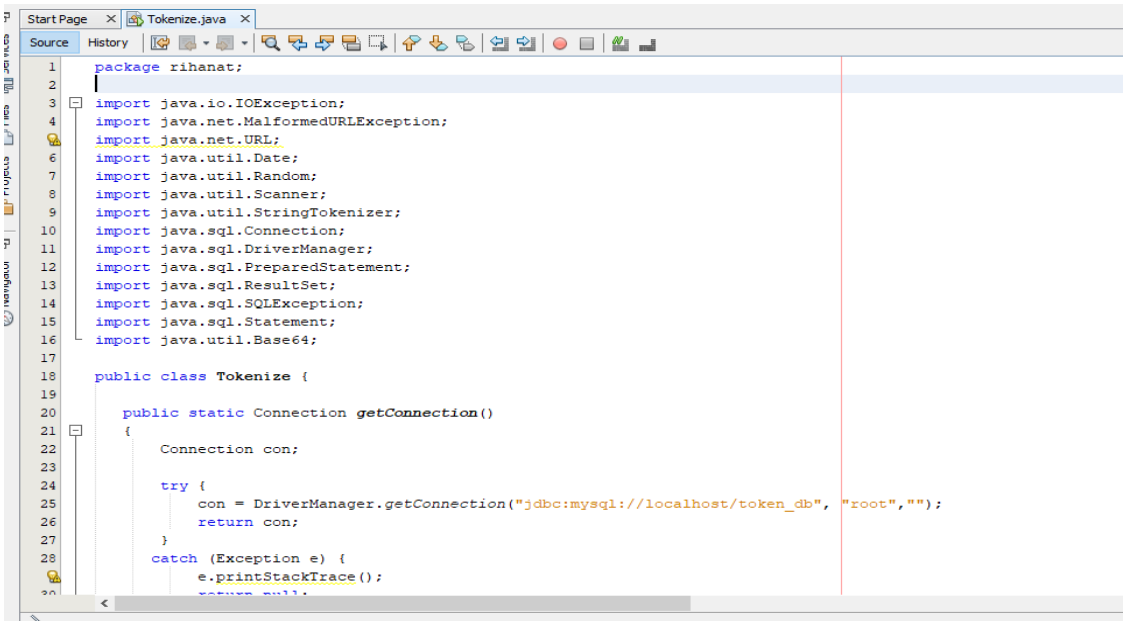
IMPLEMENTATION

Tokenization Implementation set up include Desktop a 64bits operating system, window 10pro version 20H2, Intel(R) Celeron (R) CPU N3350@1.10GHz and 4GB installed RAM. NetBeans 8.1 IDE, Java Platform SE binary (jdk-8u91-windows-x64) and xampp-windows-x64-7.3.7-0-VC15, Apache and MySQL, programming language that convert sensitive data into tokens and generated token, course code, result and the token ID that will be used to map the sensitive data from service provider's server are encrypted using AES 256 and stored in the server 127.0.0.1 a localhost phpMyAdmin Database. Implementation process was done through series of coding and its categorize in to four phases from declaration and definition phase to the result phase.

PHASE 1: In this phase declaration and definition of all classes of characters that are to be imported and needed throughout the coding process are declared using import java.as shown below.

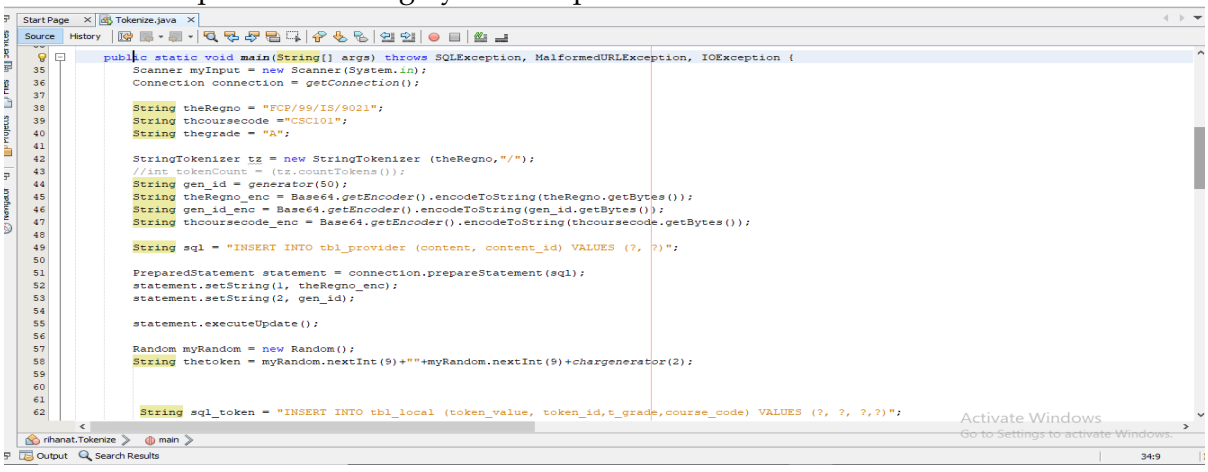
Figure: 4 Declaration Code

PHASE 2: In this phase a scanner input and a get connection code were established and used



```
package rihanat;

import java.io.IOException;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Date;
import java.util.Random;
import java.util.Scanner;
import java.util.StringTokenizer;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Base64;

public class Tokenize {

    public static Connection getConnection()
    {
        Connection con;

        try {
            con = DriverManager.getConnection("jdbc:mysql://localhost/token_db", "root","");
            return con;
        }
       catch (Exception e) {
            e.printStackTrace();
```

for calling input data in the process, followed by the tokenization process where 50 characters were counted and randomly generated to avoid duplication as shown in Fig. 4 and 5 respectively. In this work only the student registration number is tokenized therefore, the token,
course code and result remains in the data vault all encrypted while the original reg number is with service provider in a highly secured place.



```
public static void main(String[] args) throws SQLException, MalformedURLException, IOException {
    Scanner myInput = new Scanner(System.in);
    Connection connection = getConnection();

    String theRegno = "FCP/99/IS/9021";
    String thcoursecode ="CSC101";
    String thegrade = "A";

    StringTokenizer tz = new StringTokenizer (theRegno,"/");
    //int tokenCount = (tz.countTokens());
    String gen_id = generator(50);
    String theRegno_enc = Base64.getEncoder().encodeToString(theRegno.getBytes());
    String gen_id_enc = Base64.getEncoder().encodeToString(gen_id.getBytes());
    String thcoursecode_enc = Base64.getEncoder().encodeToString(thcoursecode.getBytes());

    String sql = "INSERT INTO tbl_provider (content, content_id) VALUES (?, ?)";

    PreparedStatement statement = connection.prepareStatement(sql);
    statement.setString(1, theRegno_enc);
    statement.setString(2, gen_id);

    statement.executeUpdate();

    Random myRandom = new Random();
    String thetoken = myRandom.nextInt(9)+""+myRandom.nextInt(9)+chargenerator(2);


    String sql_token = "INSERT INTO tbl_local (token_value, token_id,t_grade,course_code) VALUES (?, ?, ?,?)";
```

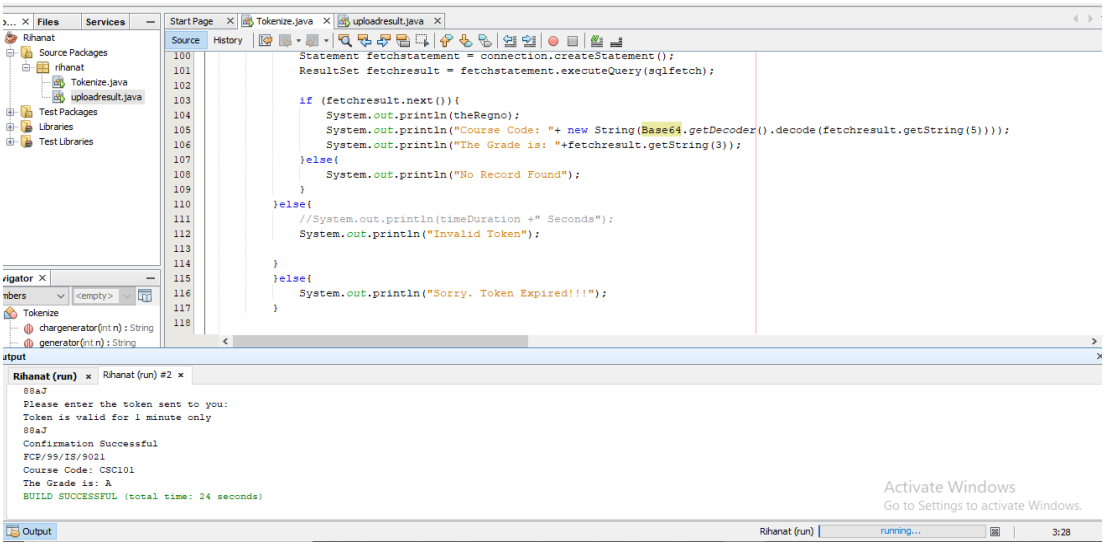
Figure: 5 Tokenization

PHASE 3: In this phase, for the fact that token is meaning less does not mean that it should be exposed. Therefore, after tokenization process the course code and token left in the data vault will be encrypted using:

String theRegno_enc = Base64.getEncoder().encodeToString(theRegno.getBytes());
String gen_id_enc = Base64.getEncoder().encodeToString(gen_id.getBytes());
Stringthcoursecode_enc=Base64.getEncoder().encodeToString(thcoursecode.getBy())

PHASE 4: In this last phase which is the results phase (i.e. Output) a token that is case sensitive will be sent to client on request to ascertain its authentication and is requesting the client to send the token back to service provider before access can be granted within 1 minute, if sent incorrectly it will return invalid, if duration exceeded 1 minute, the process will return token expire otherwise it will print confirmation successful as shown below:

Figure 6: The Result (Output)



DISCUSSION OF THE RESULT

When evaluating the efficiency performance of tokenization process as to further validate findings from this study since the security aspect has been taking care of, we consider comparing tokenization process with other security systems i.e. the three common cryptography techniques which are Encoding (Base64), Hashing (SHA-2) and Encryption (AES 256bits) on the same data length (Student Registration Number) at the same time in terms of Speed, Memory Space Consumed, Block Size and Key Length.

Speed: Figure 6, shows the result generated in nanoseconds and later converted to microseconds at the end runs of experiment carried out in terms of speed between tokenization process and other security systems which shows that tokenization execution time is the lowest which makes it the fastest compare with encoding, hashing and encryption.

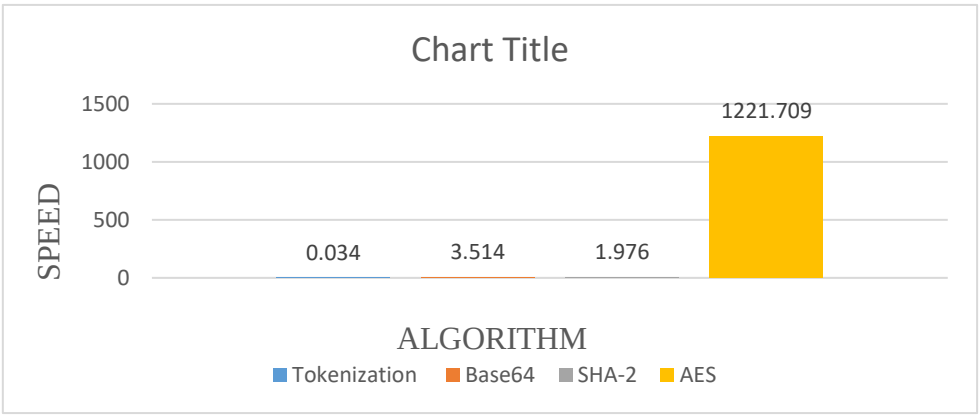


Figure 7: Speed Performance

Memory Usage: Figure 7, shows that tokenization has the least memory consumption rate while AES take highest memory usage followed by SHA-2 and Base64 respectively from the experiment carried out to determine which of the security systems consume more memory space in MB. Since tokenization only store tokens in the database and move the original data to the service. provider this will actually make it to consume less hard ware which implies that tokenization is cost effective that also better the efficiency of the security of sensitive data

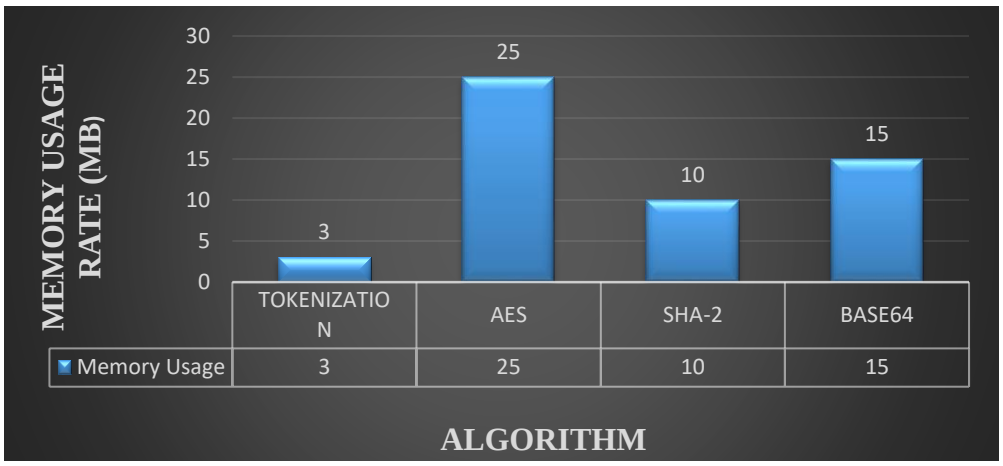


Figure 8: Memory Usage Performance

Block Size: Figure 8, the result of experiment carried out on comparison of tokenization process with other security process in terms of block size which is the output size.it shows that AES has highest block size while tokenization has the lowest. A change of one bit in plaintext leading to significant change in bits of output information AES uses a substitution permutation using multiplicative inverse and affine transformation over a gliosis field leading to high mixing of information leading to high diffusion in output.

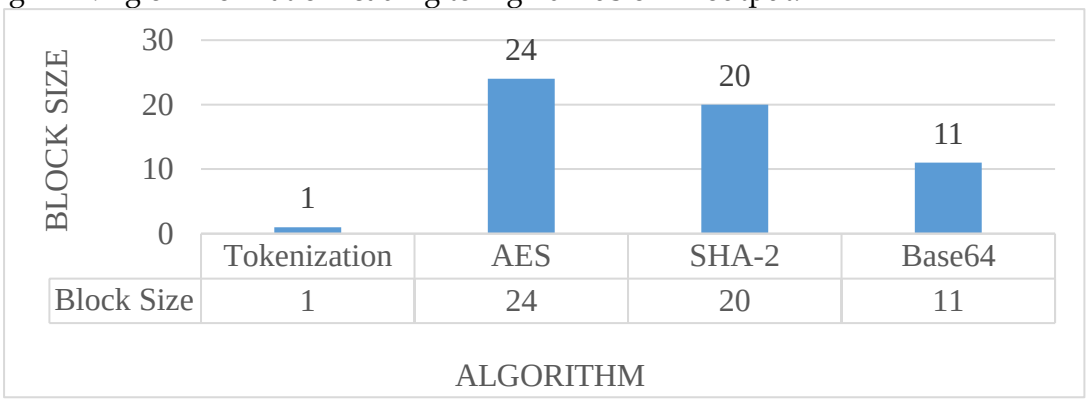


Figure 9: Block Size Performance

Key Length: Figure 9, shows that the result of encrypting the same size of data varies proportional to the size of data file. From the results it can be find that AES has the highest key size and tokenization has the lowest. Encryption different key size of the algorithms are not the same.

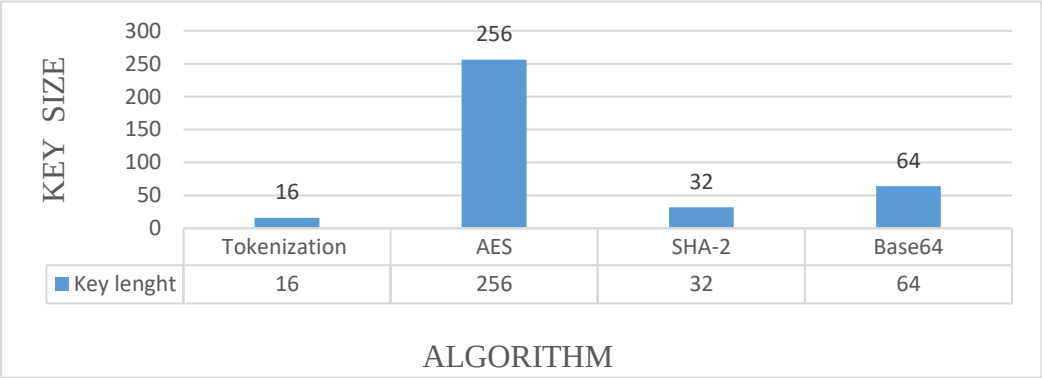


Figure 10: Key length

Table 1: Summary of Comparison between Tokenization and other Security Systems

ALGORITHM	SPEED	MEMORY USAGE (MB)	BLOCK SIZE	KEY LENGHT	SECURITY
TOKENIZTION	0.034	3	1	5	High
AES	1221.709	25	24	128	Low
SHA-2	1.976	10	20	64	Very low
BASE64	3.514	15	11	64	Very low

CONCLUSION

The main aim of this research was not to override the importance and use of data encryption in protecting sensitive data in the database but to:

1. Introduces tokenization as additional measure for better security protection on sensitive data while data encryption is still in use.
2. Though in tokenization, token has no value or relationship to the original data (i.e. irreversible) as it is in data encryption but still need to be encrypted and not to be expose to hackers.

REFERENCES

Alexander S. Gillis,(2020) Technical Writer and Editor 2020.

Bhojaraju Gunjal,(2003) Database System: Concepts and Design National Institute of Technology Rourkela.

Babich, V., and Hilary, G. 2020. Distributed Ledgers and Operations: What Operations Management Researchers Should Know About Blockchain Technology.

Fulmer, (2020).5 Ways tokenization can improve database security.

GBHackers (2019). Five Key Capabilities to Look for in Web Application Firewall Provider on Security.

Geomerchant team (2017). What is tokenization and how does it work

Hrga, A., Capuder, T., and Žarko, I. P. 2020. Demystifying Distributed Ledger Technologies: Limits, Challenges, and Potentials in the Energy Sector, IEEE Access (8).

Jenn Fulmer, (2020).5 Ways tokenization can improve database security.

Kristen Noris (2017). Encoding, Hashing and Encryption: What is the difference

Kiramat ullah. (2019). Comparison of Various Encryption Algorithms for Securing Data.

Mubina M. and Trisha P. (2016). Database Security - Attacks and Control Methods cmpica, Charotar University of Science & Technology (CHARUSAT).

Nosina Krishna Chaitanya (2020). Simple And Efficient Data Encryption Algorithm PBR Visvodaya Institute of Technology and Science College

Nxumalo, P. Tarwireyi, M.O. Adigun (2016). Towards privacy with tokenization as a service Department of Computer Science University of Zululand Empangeni, South Africa.

PCI Security Standards Council (2015). Tokenization Product Security Guidelines.

Ramzi A. H. (2017). C2 Secure Database Management Systems – a comparative study Lebanese American UniversityP.O. Box 13-5053 Beirut, Lebanon

Roger Heines (2021). The Tokenization of Everything: Towards a Framework for Understanding the Potentials of Tokenized Assets. University of St.Gallen, St.Gallen, Switzerland roger.heines@bei-sg.ch.

Sandra, D., Lil M. R. and D. Chakraborty (2014). A Cryptographic Study of Tokenization Systems Department of Computer Science,

Sam Ingallis (2021). Database Server Definition

Satyam A. (2019). Application Architectural Review on Database Security, Ethical Hacking Training-Resources INFOTECH.

Securosis, L.L.C (2018). Understanding and Selecting a Tokenization Solution.

Shadi R (2015). A Comparison of Data Encryption Algorithms with the Proposed Algorithm: Wireless Security. Faculty of Information Technology Isra University P.O. Box 22.

Shtybel, U. (2019). "A New Era of Private Securities: Application of Blockchain in Private Capital Markets Infrastructure," Journal of Digital Banking (4:2), pp. 152–160

Tulane (2021). What is Cryptograph and Benefit. Tulane School of Professional.