HASIL CEK_85 Database

by 85 Database Jurnal Dosen

Submission date: 28-May-2022 09:53AM (UTC+0700)
Submission ID: 1845719890
File name: 85 Database.pdf (1.24M)
Word count: 4032
Character count: 22012

Jurnal Mantik, 5 (4) (2022) 2127-2135
Published by:Institute of Computer Science (IOCS)
Jurnal Mantik
Journal homepage: www.iocscience.org/ejournal/index.php/mantik/index

Database Forensics in Software as A Service Service using Stored Procedure

Rusydi Umar[1], Imam Riadi[2], Purwanto[3]

[1,3]Program Studi Teknik Informatika, [2]Program Studi Sistem Informasi
[123]Universitas Ahmad Dahlan

E-mail: rusydi@mti.uad.ac.id

ARTICLEINFO

Article history:

Received: Des 18, 2021
Revised: Jan 21, 2022
Accepted: Feb 30, 2022

Keywords:

Stored Procedure;
SQLI;
SQLIA;
Encryption;
Decryption

ABSTRACT

Recently, the use of web-based applications has increased significantly, especially online services, most of which are used for digital transaction activities that use the internet. However, the increasing use of online services often ignores the privacy and security aspects of an application, web developers making it an attractive target for security concerns. In this article, the proposed countermeasures include two mechanisms, namely: the use of stored procedure parameters, and the use of stored procedure encryption in SQL server. The goal is to prevent the dangers of internet crime attacks using structured query language injection attacks (SQLIA). In addition, an analytical evaluation of prevention and protection should also be carried out. The solution offered is prevention and protection using stored procedures because it can prevent SQLI attacks effectively and efficiently.

1. Introduction

Currently, the use of the internet has significantly increased in the dissemination of information and various other online transactions by creating the wheel of the informatics revolution in recent years [1], [2]. The increasing use of the internet or web-based applications for most activities in digital transactions raises security problems [3], [4].

Web applications that use database-based content are ubiquitous. Companies whose business models focus on the internet, such as Google, Amazon, Microsoft, Oracle are real examples. Almost every large company using the web uses relational databases [5], [6]. Typically, web users provide information such as usernames and passwords, and in return receive customized content. This database includes sensitive information, such as customer data [7], [8]. Thus, the Internet becomes a widespread information infrastructure.

Since the advent of web programming, web applications have become a fast way to offer access to online services via the internet [9]. This causes web-based applications to gain immense popularity in the world because; Web applications have integrated enterprise integration by allowing many Internet-capable applications [10], [11].

Web applications usually interact with databases via the backend, wherein, the data used in web applications often contain sensitive and confidential information [12], [13]. Web applications are often vulnerable to attack due to lack of design, misconfiguration, or weakness of the written code of web applications [14], [15]. Structured query language injection (SQLI) attacks take advantage of the trust that exists between the user and the server owner. Some input or output validation features are not present on the server to reject malicious code [16]. However, it is important to provide web application protection from targeted SQLIA [17]. In addition, the SQLIA instruction detection framework carried out by Yassin et al., 2017 to provide a high degree of portability in SaaS (Software as a Service) cloud provider applications in a service-based manner to enhance security [18].

To maintain message security, a system called cryptography is needed. Cryptography has become a science that is widely used to maintain information security with mathematical calculation techniques. This

Accredited "Rank 4" (Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

technique can convert plain text using keywords into random messages or encrypted text [19]. Cryptography uses two algorithmic rules or methods, namely encryption and decryption. This study uses standard encryption available on SQL server stored procedures, while decryption uses software from a third party (third party).

Companies without using technology will surely be left behind in many aspects such as efficiency, connectivity, effectiveness and productivity. The internet can be obtained by searching for the desired information. Connection from the network is necessary under any circumstances, but connectivity does not always work well, there are many complications or problems related to connections that do not work well. The rapidly increasing penetration of internet and computer networks in addition to providing convenience, also has security problems for companies and individual database users [20], [21]. The current research focuses on the vitality and importance of SQL injection attack problems in stored procedures to detect and prevent these attacks in web-based application environments through SQLIA prevention and protection measures. The proposed countermeasures include four mechanisms for dealing with web-based applications or web sites from SQLIA in the use of stored procedures.

2. Method

This research method uses 3 (three) stages, namely; hardware identification and software identification, design, implementation and analysis as shown in Figure 1.

Fig 1. Research Method

Keywords are scientific terms that are often encountered in the field of science related to articles, and can be used as search keys to find articles. Use 3 -5 words or phrases/phrases that are key to the submitted article.

2.1 Identify Hardware and Identify Software

At this stage, identification of the required hardware and software is carried out. The hardware and software specifications used can be seen in table 1.

Table 1.
Hardware and Software

Product	Version
Komputer	
Microsoft Windows	10
SQL Server	2012
SQL Server Management	18.2
ASP.NET	2017
C#NET	2017
Encryption Tools	18.2
Decryption Tools	Trial

The SQL server used in this study uses two versions, namely: SQL 2000 and SQL 2012. Meanwhile, we chose SQL Server Management Studio to use version 18.5 because only that version can access SQL server 2000 and SQL server 2012 or versions above. Decryption tools using the following software: SQL Decryptor and DB Foge SQL Decryptor. Both software use a trial version and can be downloaded from the internet.

Accredited "Rank 4" (Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

2.2 Design

Technical design needs to be done at the experimental stage. The mechanism includes the creation of stored procedures, encryption, decryption, and parameterized queries using input validation [22]. There are several things that need to be prepared before conducting a forensic analysis of digital encryption and description as follows:

2.3 Data Design

This experiment or research uses two SQL query data as follows: the first does not use a stored procedure and the second uses a stored procedure [23]. There are two types of stored procedures: user defined stored procedures and system stored procedures. The user defined stored procedure is used by the user to create SQL statements such as select, update, or delete records from the database, while the system stored procedure is used by the system. The difference between a stored procedure and an SQL query without a stored procedure is the use of commands such as create and alter procedures such as program code 1 and 2 below.

Program Code 1. Stored Procedure

```
Input : Stored Procedure


Create Procedure [dbo].[sp_check_2] @TC nvarchar(100)

AS

BEGIN

;With cte ([TC NUMBER],[DATE],[CUSTOMER],[FILED
ENGINEER],[ACTIVITY HISTORY],[STATUS])
```

Program Code 2. No Stored Procedure

```
Input : Tanpa Stored Procedure


Select tbl_CallActivity.CallRemarks As
TC_NUMBER,UPPER(tbl_CallActivity.ActivityDate) from Check
```

2.4 Encryption Design

The encryption function is already available on SQL server. Its use is enough to call the function in the use of stored procedures using a query or command encryption at the time of making the stored procedure [24], [25]. Furthermore, the encryption process can be carried out after being successfully encrypted, the next step can be tested by reading the file or querying the stored procedure using plain text (notepad), whether it is read or not. If the data displayed is in the form of strange characters / not read at all, it means that the encryption process was successful [26], [27]. Listing encryption can be seen in program code 3 below.

Program Code 3. **encryption**

```
Input : Encryption

with encryption

AS

BEGIN

;With cte ([TC NUMBER],[DATE],[CUSTOMER],[FILED ENGINEER],[ACTIVITY HISTORY],[STATUS])
```

2.5 Decryption Design

Decryption using third party software (third party). The software used is software to decrypt stored procedures using SQL decryptor. The SQL decryptor used is a trial version and can be downloaded on the internet. Make sure before using the SQL decryptor tool, you should backup the master database first to guard against unwanted things. After a successful backup, decryption is carried out using a stored procedure. The result is that if the source of the stored procedure query can be re-read the query using plain text (notepad), it means that the decryption is successful.

Figure 1 provides an overview of the proposed security to protect web-based applications or websites from SQLIA hazards in stored procedures. Protective prevention includes four methods or mechanisms in preventing attacks. This mechanism is carried out to protect web-based applications from four types of attacks, namely: bypass attacks with dynamic queries, stored procedure attacks using parameters, stored procedure attacks using encryption, and stored procedure attacks using decryption stored in SQL server.

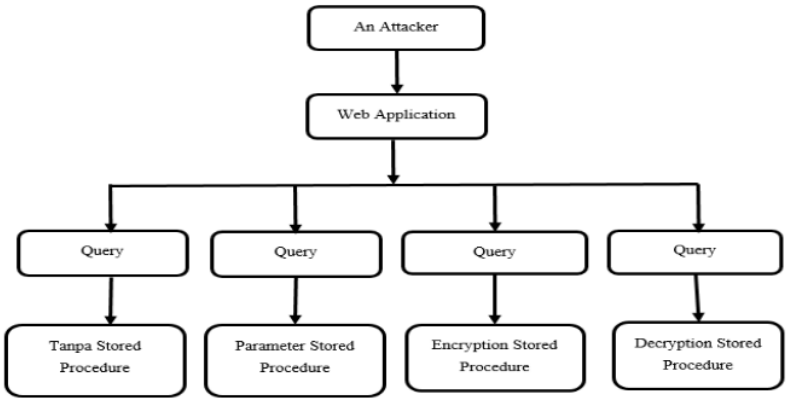

Fig 2. Attack Method

2.6 Implementation and Analysis

This experiment uses the C# programming language which is implemented and tested using the ASP.NET infrastructure. As a precaution we use three mechanisms or methods to prevent SQLIA in stored procedures such as: using parameters, using encryption, and using decryption in stored procedures.

This study uses a SQL server database version 2012 and SQL server management studio using version 18.5. One of the advantages of version 18.5 is that it can manage older versions of databases such as SQL server 2000 and the latest SQL server. So it is quite capable in managing all versions of SQL server databases.

This type of attack can be implemented through sending malicious SQL statements in the form of SQLI by the attacker to gain illegal access to the backend database [28], [29].

3. Result and Analysis

3.1 Query Implementation

The next stage illustrates the procedure of the prevention technique or method in each of the mechanisms mentioned in Figure 2. To evaluate our proposed prevention, we conducted an experiment using two scenarios by attacking the BIS (Business Information Solution) website, and this application provides many services. such as looking for TC (Test Case), SC (Store Code), and summary sales data. Figure 3 website application (A) does not use a stored procedure and (B) uses a stored procedure.

Accredited "Rank 4" (Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

Fig 3. Websites Injected and Protected

The first scenario using an injected (A) website is vulnerable to various types of SQLIA attacks. Vulnerable sites include loopholes in dynamic queries. The second scenario using a protected website (B) has been protected using parameters in the stored procedure.

The third scenario uses a protected website (B) stored procedure using encryption, and the fourth scenario uses a protected website (B) stored procedure that has been encrypted after decryption [30].

3.2 First and Second Scenarios

The next stage is to carry out an attack using the first scenario (Legitimate Query – Data Transfer) using SQL queries without using stored procedures and the second scenario using parameterized stored procedures (Bypass Attacks - Parameterized Stored Procedure). The steps of the attack can be seen in Figure 4. An example of the first scenario is an attack using SQL select queries or it can also be called SQLI. While the second scenario is to perform SQLI attacks using stored procedures and parameters. If the parameters do not match then the query will be stopped.

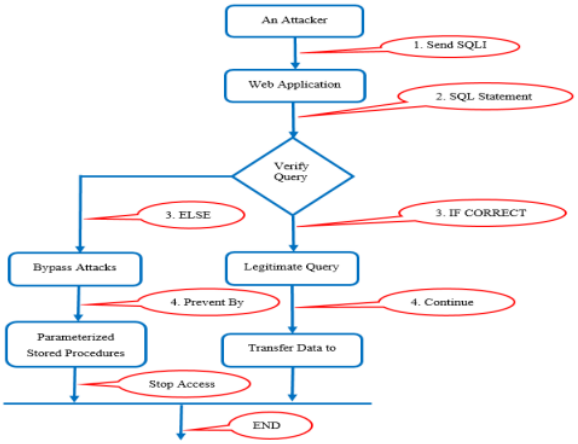

Fig 4. First and Second Scenarios

a. Send SQLI: When an attacker sends malicious code to implement an injection attack on a website, as shown in Figure 5, the attacker can enter any query and add a query that is always denoted by the value true as a result of the syntax on the SQL server, for example when the attacker places the word: TC-SC-14-0002598' or 1=1--, then the attacker can get all TC/SC numbers.

Accredited "Rank 4" (Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

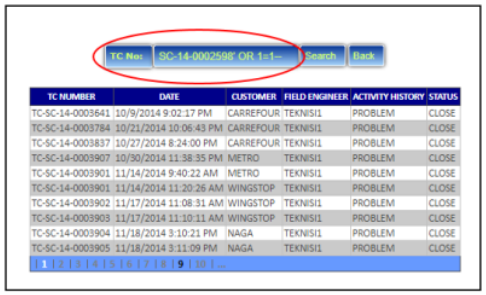

Fig 5. SQLI Without Stored Procedure

Figure 6 SQLI using a stored procedure shows that the results displayed are not found. Seen the difference in the results of the two web-based applications, which use stored procedures have better security.

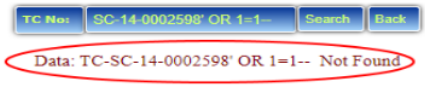

Fig 6. SQLI using a stored procedure

b. SQL Statement: After the attacker sends malicious code, the web-based application will create a malicious SQL statement or error query which will be checked in the next step. As shown in Figure 4. Searching with prepositions using dynamic queries can be implemented with the following query: Select * from BIS_Table where prepositions = '" + TextBox1.Text + "';

c. Verify Query: The query will be checked by determining if it has malicious Bypass Attacks. In the worst case, when the attacker can gain access to the database, after that the attacker gets all the sensitive information. As shown in Figure 5 SQLI without a stored procedure.

d. Query Legitimacy: Without using stored procedures but only using SQL queries such as select as shown in Figure 7.

```
protected void Button1_Click(object sender, EventArgs e)
{
    connection();
    string query = "select tbl_CallActivity.CallRemarks As TC_NUMBER,t

    SqlCommand com = new SqlCommand(query, con);

    SqlDataReader dr;
    dr = com.ExecuteReader();
```

Fig 7. Sored Procedure Parameters

3.3 Third and Fourth Scenarios

The next stage uses the third and fourth scenarios, namely the encryption stored procedure and the decryption stored procedure which can be seen in Figure 8.

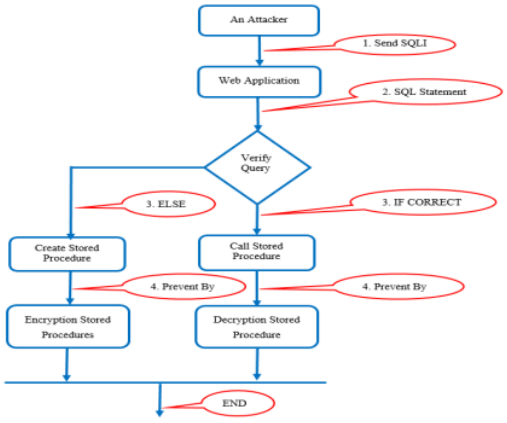

Fig 8. Third and Fourth Scenarios

Accredited "Rank 4" (Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

This scenario is the worst possible scenario, because it is almost impossible for this scenario to be attacked by external parties. To run this scenario, you need at least a database administrator user and password. This section describes the attack steps once the attacker has successfully entered the backend.

a. Attacker: An attacker managed to gain access to the source database and obtain sensitive information about the stored procedures and tables due to the insecurity of the database.

b. SQL Statement: Some system commands in the data source (SQL server) can be used by an attacker to serve the contents of a stored procedure query such as sp_helptext. Figure 9 is an example of using sp_helptext using a stored procedure: first without using encryption and second using encryption.

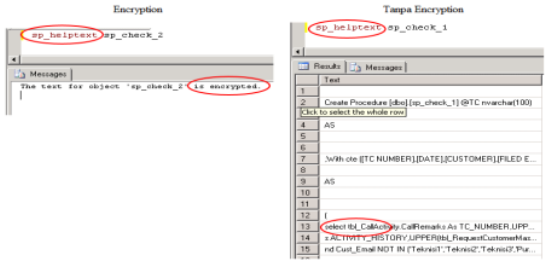

Fig 9. Encryption Stored Procedure

Encryption stored procedures protect against attackers who want to display queries or contents of stored procedures or sensitive information about tables, parameters, etc. This attack is carried out using one of the system commands such as sp_helptext. Therefore, to avoid this attack, encryption method can be used when creating a stored procedure at the programming level to ensure that an attacker cannot find out or get any sensitive information from the stored procedure. As shown in Figure 9.

Decryption stored procedures are often used by attackers using SQL decryptor (third party) tools to decrypt encrypted stored procedures. Tests were carried out using the SQL decryptor tool and of the two decryptor tools only one succeeded in decrypting. Figure 10 comparison of the results of SQL decryption tools.

Tools	Hasil
SQL Decryptor	Fail
DB Foge SQL Decryptor	Pass

3.4 Scenario Comparison Results

Based on the results of testing using four scenarios, it can be seen in Table 2. Many protection mechanisms are able to prevent some SQLIA malicious attacks. The results show that the mechanism that has been tested is effective to protect web-based applications from SQLIA in the use of stored procedures

Table 2. Comparison Result

Stored Procedure	Security
SQL Command	Fail
Parameter Input	Pass
Encryption	Pass
Decryption	Pass

Accredited "Rank 4" (Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

4. Conclusion

From the results of the tests that have been carried out, it can be concluded that the implementation of stored procedures using parameters and encryption can improve security and prevent theft of sensitive data. But from several experiments using stored procedures that have been encrypted, it turns out that decryption was successfully carried out using the SQL decryptor tool.

References

[1] J. Gondohanindijo, "Teknologi Internet Berbasis Komputer Awan (Cloud Computing)," Komputaki, vol. 1, no. 1, pp. 10–27, 2016.

[2] I. Mutia, "Jurnal String Vol . 1 No . 1 Tahun 2016 ISSN : 2527 – 9661 PEMANFAATAN KOMPUTASI AWAN (CLOUD COMPUTING) BAGI Pendahuluan ISSN : 2527 – 9661 Tinjauan Pustaka," String, vol. 1, no. 1, pp. 1–9, 2016.

[3] M. Hänninen, "Review of studies on digital transaction platforms in marketing journals," Int. Rev. Retail. Distrib. Consum. Res., no. August, 2019, doi: 10.1080/09593969.2019.1651380.

[4] D. E. W. H. W. Liana Endah Susanti, Ratna Anggraini, "the Analysis of Standard Agreement in Credit Transactions Through Financial Technology Viewed From Law No. 8 of 1999 Concerning Consumer Protection," UNIFIKASI J. Ilmu Huk., vol. 6, no. 1, p. 61, 2019, doi: 10.25134/unifikasi.v6i1.1603.

[5] G. Buehrer, B. W. Weide, and P. A. G. Sivilotti, "Using parse tree validation to prevent SQL injection attacks," SEM 2005 - Proc. 5th Int. Work. Softw. Eng. Middlew., no. May, pp. 106–113, 2005, doi: 10.1145/1108473.1108496.

[6] M. Muthuprasanna, W. Ke, and S. Kothari, "Eliminating SQL injection attacks - A transparent defense mechanism," Proc. Eighth IEEE Int. Symp. Web Site Evol. WSE 2006, pp. 22–30, 2006, doi: 10.1109/WSE.2006.9.

[7] W. G. J. Halfond, S. R. Choudhary, and A. Orso, "Improving penetration testing through static and dynamic analysis," Softw. Test. Verif. Reliab., vol. 21, no. 3, pp. 195–214, 2011, doi: 10.1002/stvr.450.

[8] N. S. Ali, "A four-phase methodology for protecting web applications using an effective real-time technique," Int. J. Internet Technol. Secur. Trans., vol. 6, no. 4, p. 303, 2016, doi: 10.1504/ijitst.2016.10003854.

[9] P. Ahluwalia, U. Varshney, K. S. Koong, and J. Wei, "Ubiquitous, mobile, pervasive and wireless information systems: Current research and future directions," Int. J. Mob. Commun., vol. 12, no. 2, pp. 103–141, 2014, doi: 10.1504/IJMC.2014.059738.

[10] S. Shrivastava and R. Tripathi, "Attacks Due to SQL injection & their Prevention Method for Web-Application," Int. J. Comput. Sciecne …, vol. 3, no. 2, pp. 3615–3618, 2012.

[11] J. P. Shim, U. Varshney, S. Dekleva, and G. Knoerzer, "Mobile and wireless networks: Services, evolution and issues," Int. J. Mob. Commun., vol. 4, no. 4, pp. 405–417, 2006, doi: 10.1504/IJMC.2006.008949.

[12] D. Abdoulaye Kindy and A.-S. Khan Pathan, "A Detailed Survey on Various Aspects of SQL Injection in Web Applications: Vulnerabilities, Innovative Attacks, and Remedies."

[13] M. Y. Kim and D. H. Lee, "Data-mining based SQL injection attack detection using internal query trees," Expert Syst. Appl., vol. 41, no. 11, pp. 5416–5430, 2014, doi: 10.1016/j.eswa.2014.02.041.

[14] A. S. Tsiaousis and G. M. Giaglis, "Mobile websites: Usability evaluation and design," Int. J. Mob. Commun., vol. 12, no. 1, pp. 29–55, 2014, doi: 10.1504/IJMC.2014.059241.

[15] N. S. Ali, A. S. Shibghatullah, and M. H. Al Attar, "Review of the defensive approaches for structured query language injection attacks and their countermeasures," J. Theor. Appl. Inf. Technol., vol. 76, no. 2, pp. 258–269, 2015.

[16] A. K. Baranwal, "Approaches to detect SQL injection and XSS in web applications," Eece 571B, Term Surv. Pap. April 2012, no. April, 2012.

[17] S. D. Ankush, "XSS Attack Prevention Using DOM based filtering API XSS Attack Prevention Using DOM based fitering API," Dep. Comput. Sci. Eng. Natl. Inst. Technol. Rourkela Rourkela – 769 008, India, 2014.

[18] M. Yassin, H. Ould-Slimane, C. Talhi, and H. Boucheneb, "SQLIIDaaS: A SQL Injection Intrusion Detection Framework as a Service for SaaS Providers," Proc. - 4th IEEE Int. Conf. Cyber Secur. Cloud Comput. CSCloud 2017 3rd IEEE Int. Conf. Scalable Smart Cloud, SSC 2017, pp. 163–170, 2017, doi: 10.1109/CSCloud.2017.27.

[19] A. Fadlil, I. Riadi, and A. Nugrahantoro, "Data Security for School Service Top-Up Transactions Based on AES Combination Blockchain Technology," Lontar Komput. J. Ilm. Teknol. Inf., vol. 11, no. 3, p. 155, 2020, doi: 10.24843/lkjiti.2020.v11.i03.p04.

[20] I. Riadi, S. Sunardi, and E. Handoyo, "Security Analysis of Grr Rapid Response Network using COBIT 5 Framework," Lontar Komput. J. Ilm. Teknol. Inf., vol. 10, no. 1, p. 29, 2019, doi: 10.24843/lkjiti.2019.v10.i01.p04.

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

[21] E. Haryanto and I. Riadi, "Forensik Internet Of Things pada Device Level berbasis Embedded System," J. Teknol. Inf. dan Ilmu Komput., vol. 6, no. 6, p. 703, 2019, doi: 10.25126/jtiik.2019661828.
[22] Sunardi, I. Riadi, and M. H. Akbar, "Penerapan Metode Static Forensics untuk Ekstraksi File Steganografi pada Bukti Digital Menggunakan Framework DFRWS," J. RESTI (Rekayasa Sist. dan Teknol. Informasi), vol. 4, no. 3, pp. 576–583, 2020.
[23] I. Zuhriyanto, A. Yudhana, and I. Riadi, "Analisis Perbandingan Tools Forensic pada Aplikasi Twitter Menggunakan Metode Digital Forensics Research Workshop," J. RESTI (Rekayasa Sist. dan Teknol. Informasi), vol. 4, no. 5, pp. 829–836, 2020.
[24] A. Wijaya, "Sistem Enkripsi Menggunakan Algoritma Aes-128 Pada Prototype Community Messenger Berbasis Android Encryption System Using Aes-128 Algorithm on Prototype Community Messenger Android-Based," vol. 2, no. 2, pp. 3306–3311, 2015.
[25] Y. Prihartono and G. Bagio, "Pengembangan Aplikasi Pengamanan File Sebagai Solusi Keamanan Data pada Smartphone Berbasis," Semin. Nas. Sist. Inf. Indones., vol. 1, no. 1, pp. 1–8, 2016.
[26] S. Rizvi, A. Kurtz, I. Williams, J. Gualdoni, I. Myzyri, and M. Wheeler, "Protecting financial transactions through networks and point of sales," J. Cyber Secur. Technol., vol. 4, no. 4, pp. 211–239, 2020, doi: 10.1080/23742917.2020.1796474.
[27] B. A. Sassani Sarrafpour, R. Del Pilar Soria Choque, B. Mitchell Paul, and F. Mehdipour, "Commercial security scanning: Point-on-Sale (POS) vulnerability and mitigation techniques," Proc. - IEEE 17th Int. Conf. Dependable, Auton. Secur. Comput. IEEE 17th Int. Conf. Pervasive Intell. Comput. IEEE 5th Int. Conf. Cloud Big Data Comput. 4th Cyber Sci., pp. 493–498, 2019, doi: 10.1109/DASC/PiCom/CBDCom/CyberSciTech.2019.00099.
[28] D. Shirkhedkar and S. Patil, "Design of digital forensic technique for cloud computing," Int. J. Adv. Res. Comput. Sci., vol. 7782, no. 4, pp. 192–194, 2014.
[29] X. Feng and Y. Zhao, "Digital forensics challenges to big data in the cloud," Proc. - 2017 IEEE Int. Conf. Internet Things, IEEE Green Comput. Commun. IEEE Cyber, Phys. Soc. Comput. IEEE Smart Data, iThings-GreenCom-CPSCom-SmartData 2017, vol. 2018-Janua, pp. 858–862, 2018, doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2017.132.
[30] J. Enkripsi, "Jenis-Jenis Enkripsi http://fajj27blog.wordpress.com/2009/01/05/jenis_enkripsi 1," pp. 1–31, 2009.
[31] Ritzkal, R. and Setiadi, D., 2021. Data Storage System Arrival and Departure Airnav Halim Perdana Kusuma Airport. Jurnal Mantik, 5(2), pp.555-562.
[32] Ritzkal, R., Prakosa, B.A. and Maulana, R.J., 2021. Human Heart Rate Detection With Web Monitoring. Jurnal Mantik, 5(3), pp.1676-1683.

Accredited "Rank 4" (Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

HASIL CEK_85 Database

ORIGINALITY REPORT

13%	13%	1%	9%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Sriwijaya University Student Paper	7%
2	iocscience.org Internet Source	1%
3	journals.ums.ac.id Internet Source	1%
4	www.readkong.com Internet Source	1%
5	ocs.unud.ac.id Internet Source	1%
6	silo.pub Internet Source	1%
7	Submitted to RDI Distance Learning Student Paper	1%
8	Submitted to Nottingham Trent University Student Paper	1%

Exclude quotes	On	Exclude matches	< 1%
Exclude bibliography	On		