Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

June 2026. Vol. 5. No. 3; e-ISSN: 2808-4519

Implementation of the AES-256 Algorithm for Securing Data Transfer on a Debian Server

Eunike Charina Ibrena Tarigan [1]*, Daniel S. Simbolon [2], Lotar Mateus Sinaga [3]

[1,2,3]*Universitas Katolik Santo Thomas*
euniketrgn27@gmail.com[1*], *danielsimbolon1213@gmail.com*[2], *lotarmateus88@gmail.com*[3]

Abstract

File Transfer Protocol (FTP) has a weakness because the data sent is not encrypted, making it vulnerable to attacks. This study aims to implement AES-256 encryption on a Debian-based FTP service using FTPS via VSFTPD and OpenSSL. The research stages include server installation and configuration, SSL/TLS certificate creation, FTP account creation, and connection testing using WinSCP. The results show that FTPS was successfully implemented on a Debian server with IP 192.168.10.2. File transfer testing ran well using TLS 1.3 and the TLS_AES_256_GCM_SHA384 cipher so that data is more secure during the transmission process. This implementation has been proven to improve data transfer security by ensuring the confidentiality and integrity of information.

Keywords: *AES-256, FTPS, VSFTPD, OpenSSL, Debian Server, TLS 1.3.*

1. Introduction

Data exchanged via computer networks forms the foundation upon which government, academia, and commercial industry pursue their objectives amidst this massive wave of digital transformation [1]. From simple text files to highly sensitive classified documents, we are constantly transmitting ever-increasing volumes of data [2]. The File Transfer Protocol (FTP) warrants special mention, as it is one of the oldest standard protocols still widely used for large-scale file distribution [3]. It has seen widespread adoption because it supports file management and offers stable, user-friendly downloading and uploading capabilities within scalable server architectures [4]. FTP was designed in its simplest form, lacking adequate security measures for data packets. When using this protocol, authentication details (usernames and passwords) and file contents are transmitted in plaintext [5].

Every data packet sent via this route carries credentials such as usernames and passwords alongside the file itself, all in plaintext. Standard FTP provides no encryption for file transfers, resulting in a very low level of network security [6]. Every data packet sent via this route carries credentials such as usernames and passwords as well as the file itself in plain text. This standard FTP protocol offers no encryption for file transfers across networks, resulting in a very low level of security [7]. Anyone with packet-sniffing software can intercept communications on a local corporate network or even a public network without accountability. To make matters worse, the completely open nature of this FTP communication channel leaves it vulnerable to disruptive cyberattacks during transmission, such as Man-in-the-Middle (MitM) attacks, which allow an attacker to alter file contents in transit. Common attacks include brute-force attempts and interception for Denial of Service (DoS) purposes [8].

To address these FTP security issues, additional security features provided by FTPS are required. FTPS establishes an encrypted connection between the client and server, ensuring that data is transferred only after encryption via the TLS protocol. This guarantees not only secure data exchange but also the confidentiality and integrity of the data throughout the process. AES-256, for instance, is a symmetric encryption algorithm that uses a 256-bit key; this makes it highly secure while remaining resource-efficient and fast. FTPS is an excellent choice for secure data transfer because it utilizes AES-256, delivering consistent performance on the stable Debian operating system [9]. Other studies indicate that this system protects digital documents in large-scale organizations with a more secure layer, particularly when combined with token-based user authentication and AES-256 [10]. Research also notes that the specific use of AES in document management systems can yield security protection that is 35% more effective than other symmetric cryptography methods [11].

AES (Advanced Encryption Standard) is a block cipher cryptographic algorithm renowned for data encryption [12]. In AES, key generation is the initial process [13]. AES supports key lengths of 128, 192, and 256 bits, making it both secure and fast. It is a block cipher operating in symmetric mode, utilizing 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys [14]. The developed core process is subsequently used for data encryption, implementing AES encryption via the Python programming language and the PyCryptodome library [15].

2. Research Methodology

2.1. Research Method

This study employs an experimental research method, specifically by building and implementing a security system for the File Transfer Protocol (FTP) service on a Debian-based operating system using the Advanced Encryption Standard (AES-256) algorithm. This method was selected to evaluate the effectiveness of the AES-256 algorithm in enhancing the security of data transfer processes on the FTP server through a series of direct implementations and tests.

2.2. Research Stages

The research stages were conducted systematically as follows:

A. Requirement Analysis
This stage involves identifying the hardware and software requirements necessary for system implementation. The software used includes Debian 12 as the server operating system, VSFTPD as the FTP server, OpenSSL for generating SSL/TLS certificates, and WinSCP as the FTP client application. Linux was chosen as the server platform due to its stability and high level of security for network services.

B. FTP Server Installation and Configuration
The next stage is the installation of the VSFTPD application on the Debian server. Once the installation is successfully completed, the FTP server configuration is adjusted to support communication via SSL/TLS, ensuring that all authentication and data transfer processes can be encrypted.

C. AES-256 Implementation via TLS
The security implementation process involves generating a digital certificate using OpenSSL and enabling the TLS protocol within VSFTPD. TLS 1.3 automatically utilizes modern cipher suites, such as TLS_AES_256_GCM_SHA384, which employ the AES-256 algorithm to encrypt data transmitted between the client and the server.
The command to generate the SSL/TLS certificate is executed using OpenSSL, specifying an RSA key length of 4096 bits and a certificate validity period of 365 days. This certificate is then integrated into the VSFTPD configuration to support the FTPS service.

D. Creation of FTP Accounts and Directories
Once the FTPS service is successfully configured, an FTP user account and a file storage directory serving as the location for data uploads and downloads are created. This step ensures that only authorized users can transfer files to the server.

E. Data Transfer Testing
Testing is conducted using WinSCP as the FTP client. The user authenticates using the created account and transfers files to the FTPS server. During the test, the connection success, the file upload process, and the server's ability to receive the transmitted data are monitored.

F. Security Verification
The final stage is the verification of communication security, performed by examining the protocols and ciphers used during the data transfer session. The implementation is deemed successful if the connection utilizes TLS 1.3 and the TLS_AES_256_GCM_SHA384 cipher, indicating the use of the AES-256 algorithm for data encryption.

2.3. Research Workflow

The research was conducted by installing and configuring VSFTPD and OpenSSL on a Debian server. Following the creation of SSL/TLS certificates and FTP accounts, connection and file transfer tests were performed using WinSCP. System security was verified by confirming the use of TLS 1.3 and the TLS_AES_256_GCM_SHA384 cipher; subsequently, the test results were analyzed to evaluate the successful implementation of AES-256-based FTPS.

2.4. Data Collection Techniques

Research data was obtained through:

a. Observation
   1. Observing the FTPS server configuration process on Debian.
   2. Observing user authentication and file transfer processes.

b. Experimentation
   1. Directly implementing AES-256 via FTPS.
   2. Testing the success of file transfers using WinSCP.

c. Documentation
Collecting data in the form of screenshots showing server configurations, file transfer processes, and verification results for the TLS 1.3 protocol and AES-256 cipher.

3. Results and Discussion

3.1. Implementation Results

This study implemented the AES-256 algorithm on a Debian FTP Server using FTPS. The implementation was carried out using VSFTPD as the FTP server and OpenSSL for generating SSL/TLS certificates. Data transfer security testing was conducted using WinSCP as the client application.

a. Debian Repository Update

At the initial stage, the system repositories and packages were updated using the following command: apt update && apt upgrade -y

The update process was completed successfully, although a warning related to the CD-ROM repository was encountered.

```
root@eunike:~# apt update && apt upgrade -y
Ign:1 cdrom://[Debian GNU/Linux 12.4.0 _Bookworm_ - Official amd64 DVD Binary-2 with fi
rmware 20231210-17:57] bookworm InRelease
Err:2 cdrom://[Debian GNU/Linux 12.4.0 _Bookworm_ - Official amd64 DVD Binary-2 with fi
rmware 20231210-17:57] bookworm Release
  Please use apt-cdrom to make this CD-ROM recognized by APT. apt-get update cannot be
used to add new CD-ROMs
Hit:3 http://deb.debian.org/debian bookworm InRelease
Hit:4 http://deb.debian.org/debian bookworm-updates InRelease
Hit:5 http://security.debian.org/debian-security bookworm-security InRelease
Reading package lists... Done
```

Figure 1: Debian Repository Update

b. VSFTPD and OpenSSL Installation

The next stage involved installing the VSFTPD and OpenSSL packages using the following command: apt install vsftpd openssl -y
VSFTPD and OpenSSL were successfully installed on the Debian server. These packages were used to establish FTPS services and support the implementation of TLS encryption.

```
root@eunike:~# apt install vsftpd openssl -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
vsftpd is already the newest version (3.0.3-13+b2).
The following packages were automatically installed and are no longer required:
  libdaxctl1 libhashkit2 libhiredis0.14 libmemcached11 libmemcachedutil2 libndctl6
  libpcre2-posix3 libpmem1 proftpd-doc
Use 'apt autoremove' to remove them.
The following packages will be upgraded:
  libssl3 openssl
```

Figure 2: VSFTPD and OpenSSL Installation

c. FTP User Creation

To restrict user access, a new FTP account named ftpuser was created using the following command: adduser ftpuser
The user account was successfully created along with its authentication password. This account was subsequently used to access the FTPS service.

```
root@eunike:~# adduser ftpuser
Adding user `ftpuser' ...
Adding new group `ftpuser' (1004) ...
Adding new user `ftpuser' (1004) with group `ftpuser (1004)' ...
Creating home directory `/home/ftpuser' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for ftpuser
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
```

Figure 3: FTP User Creation

d. Upload Directory Creation

Next, a dedicated directory was created to store transferred files using the following commands:
mkdir /home/ftpuser/upload
chown ftpuser:ftpuser /home/ftpuser/upload

```
root@eunike:~# mkdir /home/ftpuser/upload
root@eunike:~# chown ftpuser:ftpuser /home/ftpuser/upload
root@eunike:~# ls -l /home/ftpuser
total 8
drwx------ 5 ftpuser ftpuser 4096 Jun 15 07:43 Maildir
drwxr-xr-x 2 ftpuser ftpuser 4096 Jun 15 07:46 upload
```

Figure 4: Upload Directory Creation

Figure 4 shows that the upload directory was successfully created and the appropriate access permissions were assigned to the ftpuser account.

e. SSL/TLS Certificate Creation

To enable FTPS, an SSL/TLS certificate was generated using OpenSSL:
openssl req -x509 -nodes -days 365 -newkey rsa:4096

```
root@eunike:~# openssl req -x509 -nodes -days 365 -newkey rsa:4096 -keyout /etc/ssl/pri
vate/vsftpd.key -out /etc/ssl/private/vsftpd.crt
```

Figure 5: SSL/TLS Certificate Creation

Figure 5 illustrates the generation of a 4096-bit RSA private key and a digital certificate that would be used to encrypt communication between the client and the server.

f. Certificate Information Configuration

At this stage, certificate information such as the country, province, organization, and server IP address was entered as the Common Name (CN).

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ID
State or Province Name (full name) [Some-State]:Sumatera Utara
Locality Name (eg, city) []:Medan
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Universitas Katolik Santo Th
omas
Organizational Unit Name (eg, section) []:Eunike
Common Name (e.g. server FQDN or YOUR name) []:192.168.10.2
Email Address []:
```

Figure 6: Certificate Information Configuration

Based on Figure 6, the IP address 192.168.10.2 was used as the server identity in the generated digital certificate.

g. Certificate Verification

After the certificate generation process was completed, verification was performed using the following command:

ls -l /etc/ssl/private/

The vsftpd.key and vsftpd.crt files were successfully created and stored in the SSL directory.

```
root@eunike:~# ls -l /etc/ssl/private/
total 12
-rw-r----- 1 root ssl-cert 1704 Mar 12 07:14 ssl-cert-snakeoil.key
-rw-r--r-- 1 root root     2126 Jun 15 08:09 vsftpd.crt
-rw------- 1 root root     3272 Jun 15 08:07 vsftpd.key
```

Figure 7: Certificate Verification

h. FTPS Connection Using WinSCP

After the server configuration was completed, connection testing was performed using the WinSCP application.

The connection parameters used were:

- Protocol: FTP
- Encryption: TLS/SSL Explicit Encryption
- Host: 192.168.10.2
- User: ftpuser

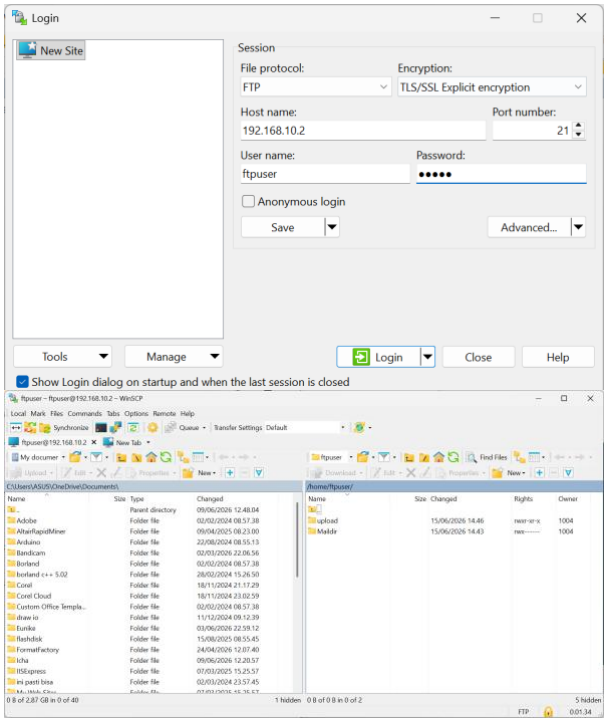

Figure 8: FTPS Connection Configuration in WinSCP

The client successfully connected to the server using FTPS with FTP and TLS/SSL Explicit Encryption settings. The server directory could be accessed properly.

i. Test File Preparation

To test the data transfer process, a text file named data_uji.txt was created containing information regarding the implementation of AES-256 on the Debian FTP Server.

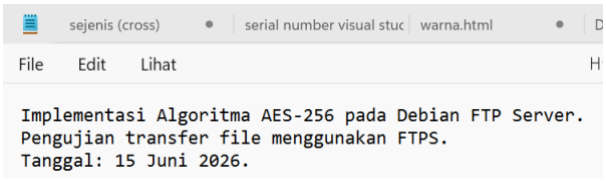

Figure 9: The data_uji.txt File

The contents of this file were used as test data for the upload process to the server.

j. Upload Directory Selection

At this stage, the file from the client computer was selected for uploading to the /home/ftpuser/upload directory. The file was then ready to be transmitted to the FTPS server.

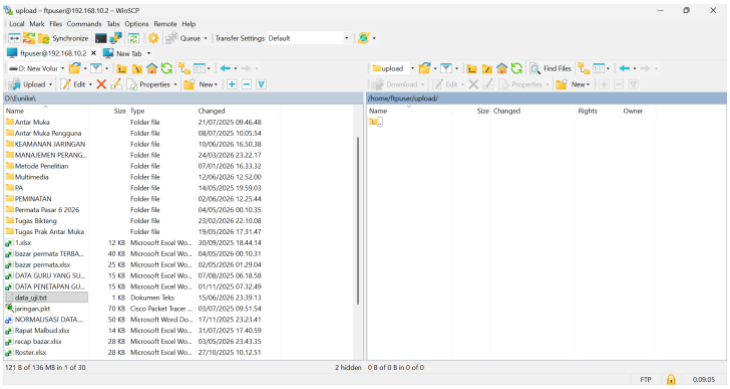

Figure 10: Upload Directory Selection

k. File Upload Process

After the upload button was selected, WinSCP displayed the destination directory on the server. The data_uji.txt file was then transferred to the upload directory that had been created previously.

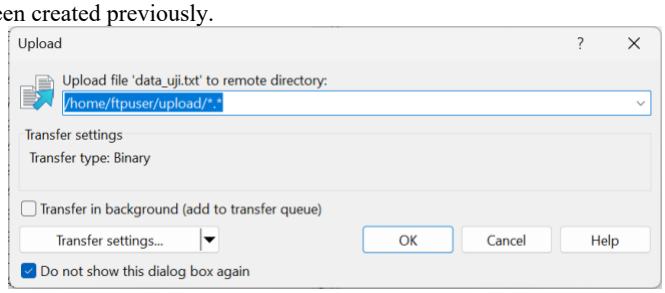

Figure 11: File Upload Process

l. Successful Upload

After the transfer process was completed, the data_uji.txt file successfully appeared in the server directory. The upload process was completed without any data transfer errors.

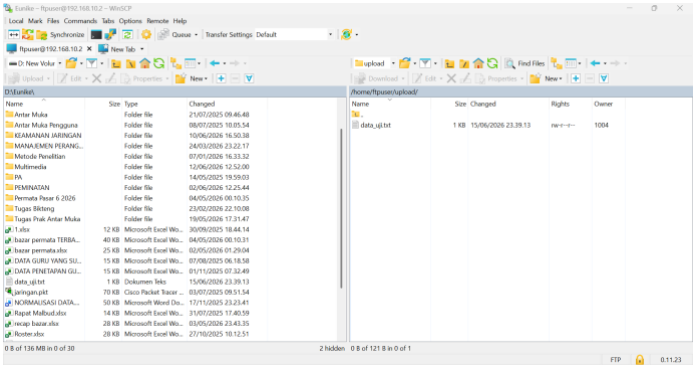

Figure 12: Successful Upload

m. File Verification on the Server

To ensure that the file had been stored on the server, verification was performed using the following command:

ls -l /home/ftpuser/upload

The data_uji.txt file was successfully stored on the server with the appropriate file size and access permissions.

```
root@eunike:~# ls -l /home/ftpuser/upload
total 4
-rw-r--r-- 1 ftpuser ftpuser 121 Jun 15  2026 data_uji.txt
```

Figure 13: File Verification

n. File Content Verification
The file contents were then examined using the following command:
cat /home/ftpuser/upload/data_uji.txt
The file content received by the server matched the file sent by the client, indicating that data integrity was successfully maintained.

```
root@eunike:~# cat /home/ftpuser/upload/data_uji.txt
Implementasi Algoritma AES-256 pada Debian FTP Server.
Pengujian transfer file menggunakan FTPS.
```

Figure 14: Content Verification

o. o. AES-256 Encryption Verification

```
Post-Handshake New Session Ticket arrived:
SSL-Session:
    Protocol  : TLSv1.3
    Cipher    : TLS_AES_256_GCM_SHA384
```

Figure 15: AES-256 Encryption Verification

The final stage involved examining the security protocol used during the data transfer process.
The results shown in Figure 15 indicate:

a. Protocol: TLSv1.3
b. Cipher: TLS_AES_256_GCM_SHA384

These results demonstrate that communication between the client and the server employed the AES-256 algorithm as the data encryption mechanism.

3.2. Discussion

Based on the implementation and testing results, the FTPS service was successfully deployed on the Debian server using VSFTPD and OpenSSL. Testing using the WinSCP application demonstrated that both the authentication process and file transfer operations could be performed successfully without any issues.
Communication security was significantly enhanced through the use of the TLS 1.3 protocol with the TLS_AES_256_GCM_SHA384 cipher suite, which implements the AES-256 algorithm to encrypt data during transmission. As a result, the transmitted data were protected against eavesdropping and unauthorized access.
Furthermore, the verification results indicated that the data_uji.txt file received by the server contained the same content as the file sent by the client. This finding demonstrates that data integrity was maintained throughout the transfer process.
Based on these results, the implementation of AES-256 through FTPS on the Debian FTP Server proved to be effective in enhancing data transfer security compared with conventional FTP, which does not employ encryption mechanisms.

4. Conclusion

Through the implementation and testing of the system, it was found that the AES-256 algorithm was successfully implemented on the Debian FTP Server service by applying FTPS (File Transfer Protocol Secure) using the VSFTPD and OpenSSL applications. With this implementation, data transfer processes can be performed securely through an encryption scheme based on the TLS 1.3 protocol.
Based on the testing results using the WinSCP application, it can be concluded that the FTPS connection was established successfully, user authentication functioned properly, and file upload and storage on the server were completed without any modification to the file contents. Security verification further demonstrated that the communication utilized the TLS_AES_256_GCM_SHA384 cipher suite, indicating the implementation of the AES-256 algorithm. Consequently, data transmitted through this communication channel, such as file uploads, were encrypted during transmission.
Therefore, the implementation of AES-256 on FTPS has been successfully tested and proven to enhance the security level of FTP services by protecting the confidentiality, integrity, and reliability of data against eavesdropping and unauthorized access during the data transfer process. This implementation can serve as an effective solution for securing data exchange in network environments based on the Debian Server platform.

References

[1] K. Saleem, M. F. Zinou, F. Mohammad, R. Ouni, A. Z. Elhendi, and J. Almuhtadi, “End-to-end security enabled intelligent remote IoT monitoring system,” no. March, pp. 1–13, 2024, doi: 10.3389/fphy.2024.1357209.
[2] R. Sharma and C. Science, “‘ Enhancing Security of Data in Cloud Computing Using A Novel Approach o f Number System ,’” vol. 02, no. 05, pp. 1–10, 2026.
[3] J. Control, P. Singh, S. V. S. Prasad, S. Upadhyay, and R. Singh, “Performance-efficient flexible architecture of m – Crypton cipher for resource-constrained applications,” vol. 1144, 2024, doi: 10.1080/00051144.2024.2395617.
[4] A. F. Ndruru, R. Rosnelly, B. S. Riza, I. Komputer, and U. P. Utama, “Analisis Perbandingan Metode Machine Learning KNN dengan Naive Bayes pada Log Serangan Jaringan,” vol. 15, 2026.
[5] R. D. Ainul, “A Lightweight Layered Security Scheme for Secure RSSI-Based Indoor Localization,” vol. 15, no. 2, 2026, doi: 10.18178/ijeetc.15.2.146-158.
[6] H. M. Mohammad and A. A. Abdullah, “Enhancement process of AES : a lightweight cryptography algorithm-AES for constrained devices,” vol. 20, no. 3, pp. 551–560, 2022, doi: 10.12928/TELKOMNIKA.v20i3.23297.
[7] M. Keh and Y. Chen, “Implementing Post-Quantum Cryptography to Industrial Wireless Networks,” 2026.
[8] T. Ariyadi *et al.*, “Analisis Kerentanan Keamanan Sistem Informasi Akademik Universitas Bina Darma Menggunakan,” vol. 22, no. 2, pp. 418–429,

2023.
[9] M. Tahir *et al.*, "PENGGUNAAN ALGORITMA AES (ADVANCED ENCRYPTION STANDARD) PADA VERACRYPT UNTUK MENGAMANKAN DATA PADA PERANGKAT PENYIMPANAN," vol. 9, no. 4, pp. 6912–6918, 2025.
[10] F. Baso and N. A. L, "Implementasi Teknik Kriptografi dengan Metode AES 256 untuk Keamanan File," vol. 3, no. 3, pp. 84–87, 2024.
[11] W. Putra, M. R. Fahlevi, and A. T. Hidayat, "Implementasi Algoritma Advanced Encryption Standard Untuk Kemanan Dokumen," vol. 1, no. 2, pp. 76–83, 2023.
[12] G. Y. Pangestu, A. I. Hadiana, and P. N. Sabrina, "Kriptografi Untuk Enkripsi Ganda Pada Gambar Menggunakan Algoritma AES (Advanced Encryption Standard) Dan RC5 (Rivest Code 5)," vol. 1, pp. 25–32, 2022.
[13] D. E. Fitriani, N. Zulatifa, D. P. Anggraini, I. Ady, and S. Indonesia, "ANALISIS IMPLEMENTASI ENKRIPSI DAN DEKRIPSI MENGGUNAKAN ALGORITMA ADVANCED ENCRYPTION STANDARD (AES) PADA JAVA," no. November, pp. 57–67, 2024.
[14] M. Suwarni and J. Wahyudi, "Comparison of the DES Cryptographic Algorithm and the AES Algorithm in Securing Document Files Perbandingan Algoritma Kriptografi DES Dan Algoritma AES Dalam Pengamanan File Dokumen," vol. 2, no. 1, pp. 41–48, 2023.
[15] A. M. Fajrin, C. Kelvin, B. Owen, and B. Aji, "Perbandingan Performa dari Algoritma AES dan RSA dalam Keamanan Transaksi," vol. 5, no. 2, pp. 696–705, 2024.