Digitograph: A Mobile Digital Signatures Application for PDF file Using ED25519 and Asymmetric Encryption

Annisa Putri Wahyuni[1], Arif Bramantoro[2*], Randi Rizal[3], Souhayla Elmeftahi[4]

[1]Institute of Advanced Informatics and Computing, Tasikmalaya 46115 (Indonesia)
[2]School of Computing and Informatics, Universiti Teknologi Brunei (Brunei Darussalam)
[3]Department of Informatics, Siliwangi University, Tasikmalaya 46115 (Indonesia)
[4]Data Engineer, École Nationale des Sciences Appliquées d'Al Hoceima, 32003 (Morocco)

* Corresponding author: arif.bramantoro@utb.edu.bn

Received 12 March 2025 | Accepted 27 April 2025 | Early Access 5 May 2025

Abstract

Digital signatures have become an essential tool in the digital era, providing a secure and efficient way to authenticate and verify the integrity of digital documents. The increasing need for remote and electronic transactions has led to a surge in the development of digital signature technology. This research presents a mobile application, Digitograph, designed to facilitate the process of digitally signing PDF files using ED25519 and Asymmetric Encryption. Several processes were employed to complete the Digitograph application, including a literature study to gather information and documents related to the development process. A research framework was prepared to ensure that the processes in the study were carried out in a directed and systematic manner. The development of the Digitograph application was successfully accomplished, with significant results demonstrating improvements in security, efficiency, and ease of use for digital signatures on PDF files. The following are some key aspects of the Digitograph application's development: 1) Enhanced security, 2) Performance and efficiency, 3) User-friendliness. Digital signatures using ED25519, and Asymmetric Encryption are one of the key technological applications in modern cryptography. ED25519 offers a high level of security, efficiency, and ease of use. This method enhances data security and significantly simplifies key management to address vulnerabilities in Asymmetric Encryption. The Digitograph application ensures the authenticity and integrity of documents, providing a practical solution in the digital era.

Keywords

Digital signatures, ED25519, asymmetric encryption, mobile application, PDF security.

I. Introduction

In this era of rapid digital transformation, the need to ensure the security and authenticity of digital documents has become increasingly urgent [1],[2]. Digital signatures have emerged as a critical solution in addressing this challenge [3]. Through cryptographic mechanisms, digital signatures enable the secure and efficient authentication and verification of digital document integrity. This is particularly relevant in an environment increasingly dominated by electronic and remote transactions, where the authenticity and security of information must be protected from risks of forgery or data manipulation. The development of digital signature technology continues to accelerate in response to the growing demand for faster and more secure transactions [4].

As communication technology evolves, remote and digital interactions have become an integral part of various sectors, ranging from business to government administration [5]. In this context, digital signatures play a key role in safeguarding the validity of electronic documents involved in these transactions. The surge in remote transactions has driven the demand for digital signature systems that are easily accessible while prioritizing security. Technologies such as ED25519 [6] and Asymmetric Encryption have become highly relevant, as they offer a high level of security without compromising efficiency.

The Digitograph application was developed in response to the growing need for a digital signature solution that is not only secure but also user-friendly, particularly for PDF files frequently used in official transactions. Digitograph is designed using ED25519, a cryptographic algorithm known for its strong security in generating and verifying digital signatures. Additionally, the application employs Asymmetric Encryption, which adds an extra layer of protection by encrypting the generated digital signature. With this approach, Digitograph offers a combination of ease of use and high security, making it an ideal choice for individuals and organizations that require a digital document signing solution

Please cite this article as: Putri Wahyuni, A., Bramantoro, A., Rizal, R., & Elmeftahi, S. (2025). Digitograph: A Mobile Digital Signatures Application for PDF file Using ED25519 and Asymmetric Encryption. International Journal of Informatics and Computing, 1 (1), 41-48.

on mobile devices.

This research aims to contribute to the advancement of digital signature technology, especially in mobile use. The study will explore the technical aspects of Digitograph, from its architecture to implementation, and evaluate its performance and security. Furthermore, the research will examine the potential applications of Digitograph in various industries and scenarios, as well as assess its benefits and limitations. Consequently, the results of this study are expected to provide a reliable and efficient solution to the challenges of digital signing in the digital era while supporting the broader adoption of digital signature technology.

II. Related Work

Various recent studies have explored the use of digital signatures and cryptographic algorithms to enhance data security in different contexts. One notable work [7] investigates the application of the Advanced Encryption Standard (AES) and digital signatures to protect messages transmitted over LoRaWAN networks from sniffing attacks. In this study, the Ed25519 algorithm was employed for digital signatures to verify the authenticity and integrity of data payloads during transmission. The findings revealed that while this security method increased the overall data size (overhead), it effectively ensured data integrity and security. This research is relevant to the development of the Digitograph application, which also utilizes Ed25519 and Asymmetric Encryption to ensure the authenticity and integrity of digitally signed PDF documents, providing a secure and user-friendly digital signing solution.

Another significant area of research focuses on the broader applications of digital signatures in cryptography. Digital signatures have become a cornerstone in modern cryptography, providing essential tools for verifying data authenticity and integrity [8]. The digital signature algorithm allows two critical operations: the signing operation, which uses a private key to generate a signature, and the verification operation, which utilizes a public key to validate the signature. The digital signature process typically involves creating a hash of the document, which is then encrypted [9],[10]. This method ensures that any alteration to the document is detectable since the digital signature would no longer match the hashed document, thus making digital signatures highly effective for securing document integrity.

ED25519, a well-known digital signature algorithm from the elliptic curve cryptography (ECC) family, has gained significant attention over the past decade due to its high security and performance [11], [12], [13]. Unlike older algorithms, ED25519 offers an optimal combination of security and efficiency, making it suitable for applications that demand quick and secure digital signatures, such as mobile applications like Digitograph. Recent studies have shown that ED25519 performs efficiently even on devices with limited resources, further underscoring its utility in mobile environments [14]. The algorithm's robustness and efficiency make it a popular choice for modern cryptographic systems that require strong security without sacrificing performance.

Asymmetric encryption, a critical cryptographic technique, is also widely used to enhance security in digital communication by employing a pair of public and private keys [15]. Unlike symmetric encryption, which relies on a single key for both encryption and decryption, asymmetric encryption ensures that data can only be decrypted by the holder of the private key, providing a higher level of security. This method is especially important in systems where digital signatures are applied, as the private key is used to sign the data, while the public key verifies it. Studies over the last decade have demonstrated the effectiveness of asymmetric encryption in securing sensitive data in various systems, particularly those that involve document signing, like the Digitograph application, which leverages this approach to secure its digital signatures [16].

III. Methodology

The process began with a literature review to gather relevant information and references related to the application development, ensuring that the project was built on a solid theoretical foundation. Additionally, a structured research framework was established to ensure that each stage of the research was carried out systematically and in an organized manner. The stages of the research involved in developing this application can be seen in the diagram below.

Fig. 1. Methodology

A. Data Collection

The data collection phase the problem identification highlights the importance of maintaining the authenticity of documents such as research papers, legal documents, and certificates, which are considered vital and must be protected from theft or tampering. Document authenticity is crucial to prevent unauthorized changes or claims, as altering the content and claiming it as one's own constitutes plagiarism and intellectual theft. This research is supported by a literature review on digital signatures, ED25519, and Asymmetric Encryption, which form the foundation for the application's development.

B. Implementation of ED25519 Algorithm (Signing Process)

Ed25519 is a cryptographic algorithm that uses a private key and a public key for the creation and verification of digital signatures. It leverages elliptic curve cryptography to enhance security, offering improved protection compared to more commonly used algorithms like RSA and DSA. In this system, the private key is used to generate the digital signature, while

the public key is utilized during the verification process. The Ed25519 algorithm operates on an elliptic curve defined over the prime field \ (2^{255} - 19\). The private key is a 256-bit integer, and the public key is a 32-byte sequence, providing a balance of strong security and efficiency.

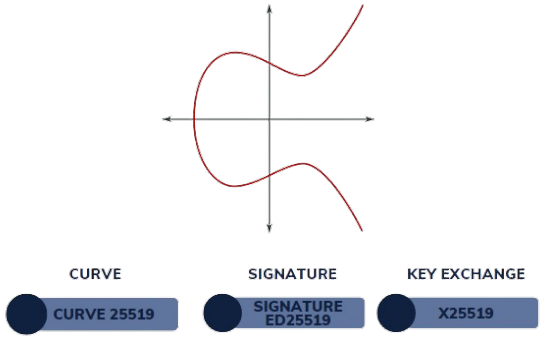


Fig. 2. Ellipse Curve

The provided code snippet demonstrates the process of creating a digital signature for a PDF file using asynchronous programming in Dart (likely within the Flutter framework). The code first reads the PDF file bytes and generates a cryptographic key pair using Ed25519. It then extracts the public key and hashes the PDF content using the SHA-256 algorithm. Afterward, the digital signature is created by signing the hash of the PDF using the private key. The signature and public key are then converted to base64 encoding. Additionally, the code prepares a QR code string containing the signature and public key data, which is later logged or used for further purposes, such as embedding in the PDF file or for verification.

C. Use of private key (Digital Signature Encryption)

In digital signature systems like Ed25519, the private key is used to create the digital signature, not to encrypt it. The private key generates a signature based on the document or message data through cryptographic functions. This signature can then be verified by anyone with the corresponding public key to confirm its authenticity. The public key is used to ensure that the signature was indeed generated by the holder of the private key. This process provides security because only the person with the private key can create a valid signature, while anyone with the public key can verify it, without needing to decrypt anything.

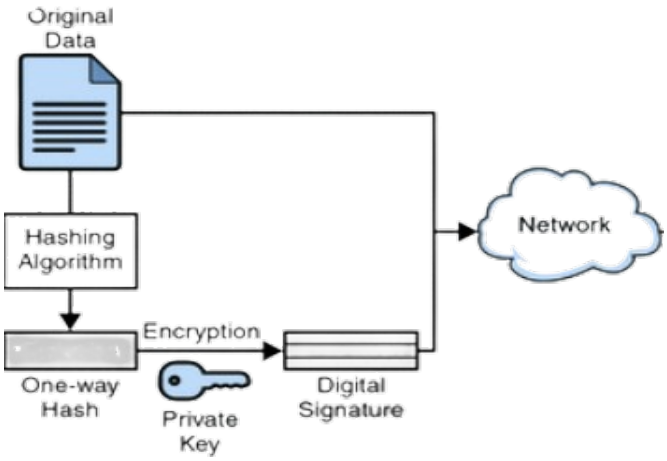


Fig. 3. Digital Signature Encryption Flow

D. Use of public key (Digital Signature Description)

The public key is used to verify that the signature was indeed generated by the corresponding private key. In systems like Ed25519, when a digital signature is created, a hash function of the document is generated and signed using the private key. During verification, the public key is used to check if the signature is valid by matching the existing hash. Thus, the public key does not "unlock" or decrypt the signature; instead, it verifies its authenticity to ensure that the signature is legitimate and originated from the private key's owner.

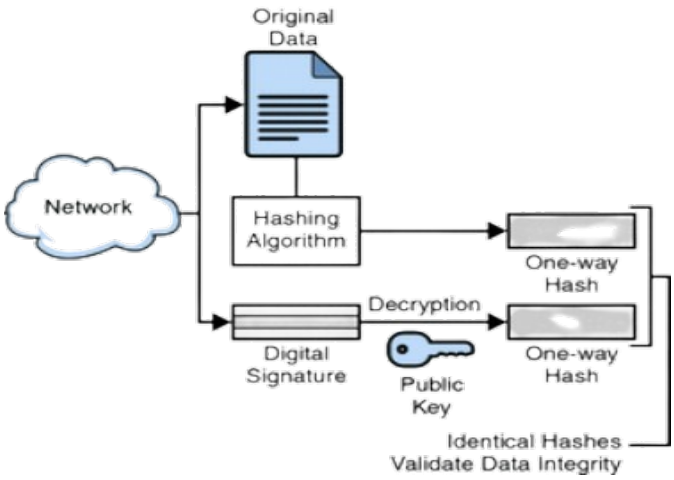


Fig. 4. Digital Signature Description Flow

E. Verification Process

The verification process in a digital signature is carried out by comparing two hashes: one from the original document (before it was signed) and one from the signed document. If both hashes match, it can be confirmed that the document remains authentic and has not been modified or claimed by someone else. However, if the hashes differ, it indicates that the document has been altered and is no longer in its original form.

IV. Result and Discussion

The development of the Digitograph application was successfully carried out with quite significant results indicating that improvements in terms of security, efficiency, and user convenience for digital signatures on PDF files. Here are some things from the development of the Digitograph application.

A. Digital Signature Flow in Digitograph Application

As explained in the methodology stage, the digital signature mechanism in the Digitograph application utilizes the Ed25519 algorithm, which employs a cryptographic pair of private and public keys during the signing and verification processes. The overall stages involve the generation of a digital signature using the private key during the signing phase and the validation of this signature with the corresponding public key during the verification phase. This ensures the authenticity and integrity of the data. The detailed steps include key generation, message signing, transmission

of the signed message, and finally, signature verification on the recipient's end.

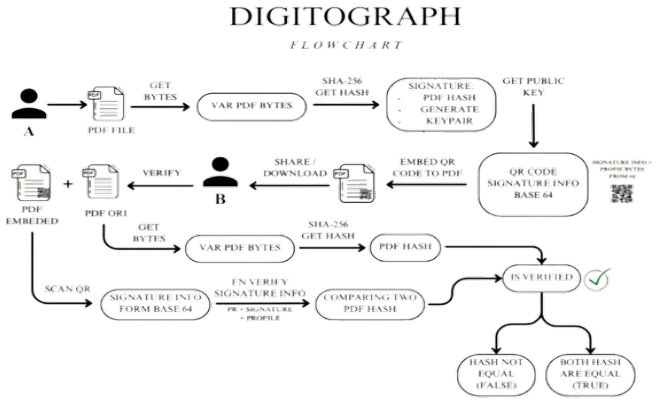


Fig. 4. The Process Flow from Signing on Digitograph

1. Signing

As depicted in Fig. 7, the signing process flow begins with User A who has a PDF file to be signed and then uploads the PDF file into the application.

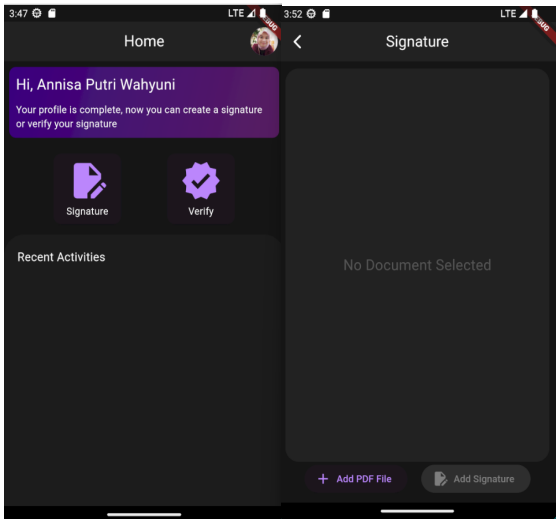


Fig. 5. Home and Signature Pages

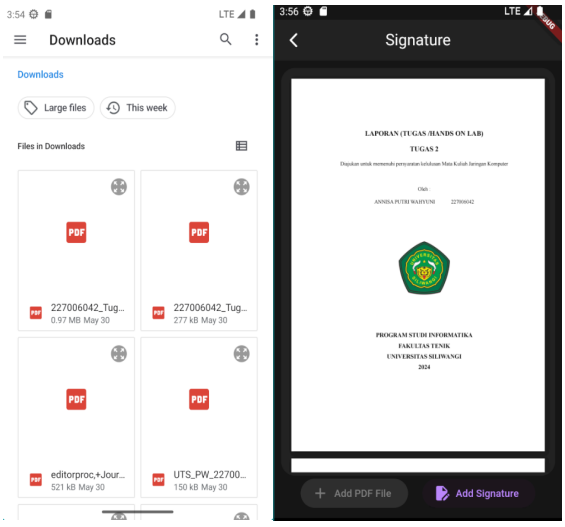


Fig. 6. Upload PDF File

This is where the algorithm begins. The PDF file is first read as a sequence of bytes, after which it undergoes a hashing process using the SHA-256 algorithm to generate a unique hash. This ensures data integrity and a secure foundation for the digital signature process.

```
final pdfBytes = fileBytes;
final pdfHash = await Sha256().hash(pdfBytes);
```

Fig. 6. Source Code Hashing

The hash result of the PDF will be input into the signature function along with the generated key pair to create both the private and public keys. Once the signature is successfully generated, the private key is used to sign the hash, while the public key is extracted from the signature. This public key can then be used later to verify the authenticity of the signature. By doing so, the system ensures both the integrity of the document and the identity of the signer during the verification process.

```
final signature = await _dt.algorithm.sign(
  pdfHash.bytes,
  keyPair: keyPair,
);
final publicKey = await keyPair.extractPublicKey();
```

Fig. 7. Signature and Public Key Extraction Functions

After the public key information is obtained, both the public key and the digital signature will be converted into byte code. To ensure that the signature and public key are readable as a String format, the base64 encoding function will be applied. This conversion makes it easier to store, transmit, and verify the signature while maintaining data integrity.

```
createQRCodeFromSignature({
  required Uint8List pdfBytes,
  required String qrContent,
}) async {
  var imageBytes = QRImage(
    qrContent,
    backgroundColor: img.ColorUint8.rgb(255, 255, 255),
    size: 300,
    errorCorrectLevel: QrErrorCorrectLevel.H,
  ).generate(); // QRImage
  final directory = await getApplicationDocumentsDirectory();
  final fileResult = await File('${directory.path}/qr-${_dt.rxFilePickedResult.st!.files.single.name}.png')
      .writeAsBytes(img.encodePng(imageBytes));

  final qrImageBytes = await fileResult.readAsBytes();
```

Fig. 8. Source Code Create a QRCode

The signature and public key, once obtained as a string, are then used to generate a QR code by encoding the string information. This QR code is embedded into the PDF file, creating a new version of the document with the signature information attached in the form of a QR code. Once the signing process is complete, the updated PDF can be saved or shared as required. This ensures both authenticity and easy verification of the signature through the QR code, enhancing document security and traceability as shown in Figure 9, 10.

```
final fusion.PdfDocument document = fusion.PdfDocument(
  inputBytes: pdfBytes,
);

final Uint8List imageData = qrImageBytes;

final fusion.PdfBitmap image = fusion.PdfBitmap(imageData);

final height = document.pages[0].graphics.size.height;

document.pages[0].graphics.drawImage(image, Rect.fromLTWH(20, height - 70, 60, 60));

List<int> bytes = await document.save();
```

Fig. 9. Source Code Embed QRcode to PDF

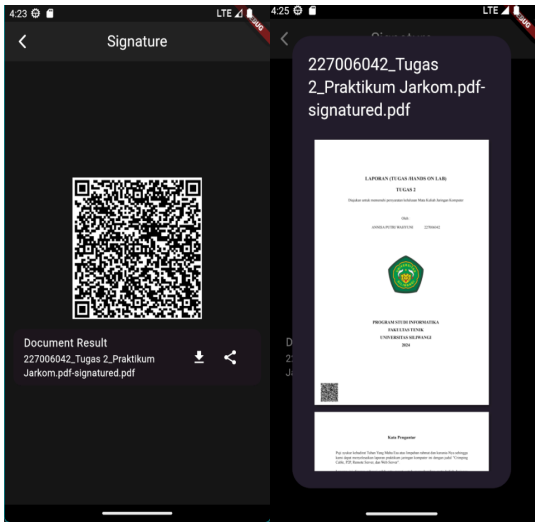


Fig. 10. PDF Embedded

b. Verify

Figure 11 illustrates the process of signature verification. The verification procedure begins with User B, who initiates the process upon receiving the signed PDF file from User A. User B proceeds to extract the embedded QR code from the PDF, which contains the signature and public key information. By decoding this data, User B can validate the authenticity and integrity of the document. The verification process ensures that the signature has not been tampered with and confirms its origin, thereby establishing trust in the document's content before further actions or approvals.

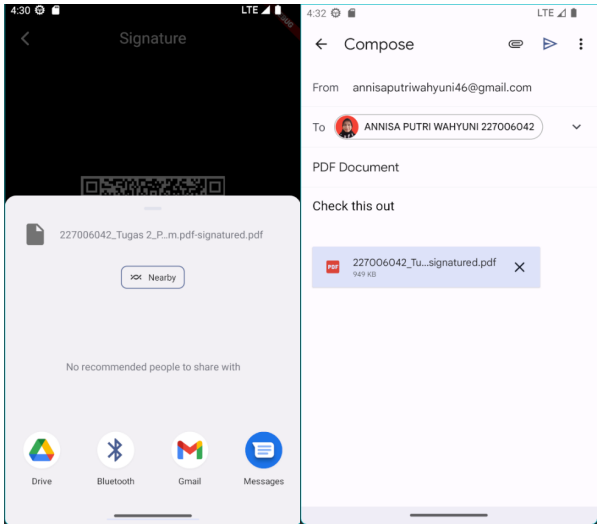


Fig. 11. Overview of the File Sharing Process

User B begins the verification process by uploading both the original PDF file, which does not contain the QR code, and the PDF file with the embedded QR code into the Digitograph application via the Verify menu. This application compares the two versions of the document, analyzing the integrity and authenticity of the QR code embedded within the signed PDF. By cross-referencing the files, Digitograph ensures that the signature, public key, and any embedded data have not been altered or compromised. This verification step provides a robust layer of security, confirming that the signed document is genuine and trustworthy.

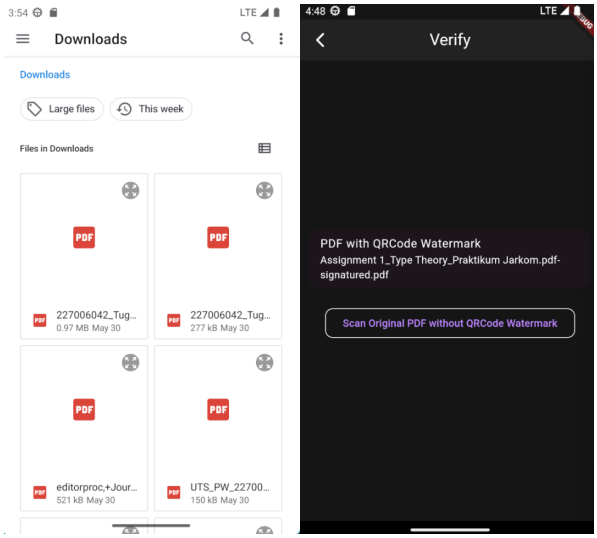


Fig. 12. Upload file to Menu Verify

For Original PDFs that do not have a QRCode, a read as bytes process is carried out so that the PDF will be read as bytes. Then a Hashing process is carried out using SHA-256 to obtain the hash value of the PDF.

```
final pdfBytes = fileBytes;
final pdfHash = await Sha256().hash(pdfBytes);
```

Fig. 13. Hashing PDF

Meanwhile, for PDFs that have a QRCode, a QRCode Scan process will be carried out. After the scan process is complete, the value of the QRCode will be obtained in the form of a string with random characters. The string will be extracted into bytes which will then be obtained as a value in the form of a signature and public key. This is where the verify process is run. The PDF hash, signature, and public key that have been obtained will be entered into the verify function.

```
Future<void> verifySign({required String qrDataString}) async {
  if (qrDataString.contains('|')) {
    final parts = qrDataString.split('|');
    final signatureBytes = base64Decode(parts[0]);
    final publicKeyBytes = base64Decode(parts[1]);
    final fileBytes = await _dt.rxPickedFile.st!.readAsBytes();

    final pdfBytes = fileBytes;
    final pdfHash = await Sha256().hash(pdfBytes);
    final signature = Signature(
      signatureBytes,
      publicKey: SimplePublicKey(
        publicKeyBytes,
        type: KeyPairType.ed25519,
      ), // SimplePublicKey
    ); // Signature
    _dt.rxIsSignatureValid.st = await _dt.algorithm
        .verify(
          pdfHash.bytes,
          signature: signature,
        )
```

Fig. 14. Signature Verification Function

In this verify function, both hashes will be compared. If both hashes are the same, then the PDF File is still proven to be authentic. Conversely, if the Hash is not the same, then the PDF File is no longer original, has been modified, or has become someone else's property.

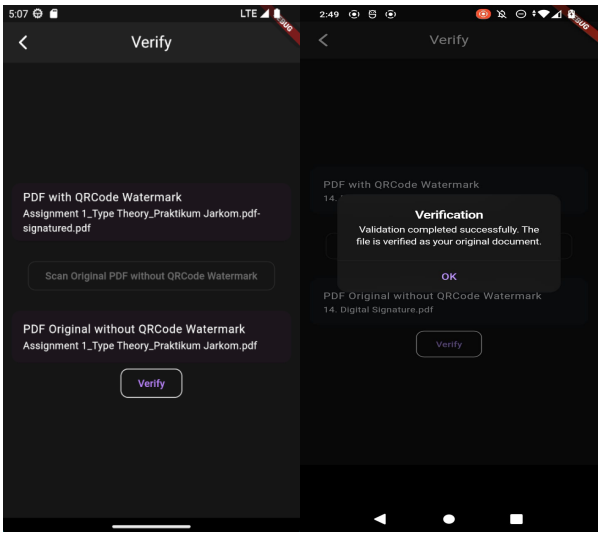


Fig. 15. Image Verification Successful

c. Testing

To ensure the success of an application, testing is needed. This testing will be carried out in several scenarios, especially in terms of signature verification such as signature verification on files that have been modified, files that have had their QRCode removed, and so on. The first test was carried out in a positive scenario, namely adding a signature to a PDF and then verifying it without carrying out any modification process on the file.

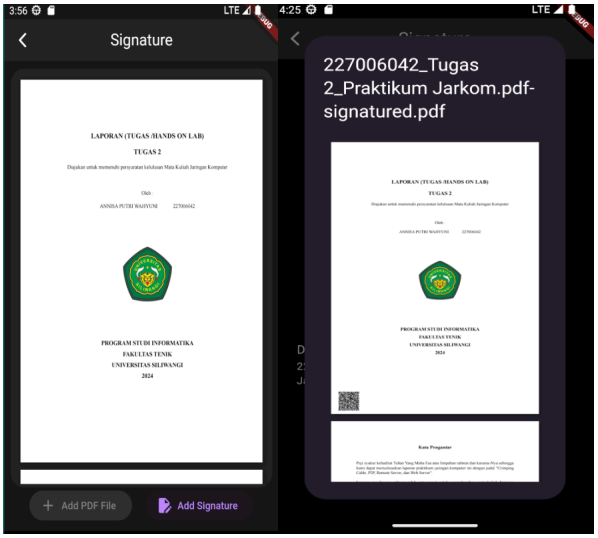


Fig. 16. Unsigned and Signed PDF file

As illustrated in Figure 17, the signature process has been successfully completed. The document is now officially signed, with a QR code visibly embedded in the lower-left corner of the first page. This QR code contains critical signature and public key information, which can be used for future verification of the document's authenticity and integrity. The positioning of the QR code ensures it is easily accessible for scanning without obstructing the document's main content. This digital signing method enhances the security of the document, offering a streamlined approach for tracking and verifying signatures. The verification process will then be carried out using the same file.

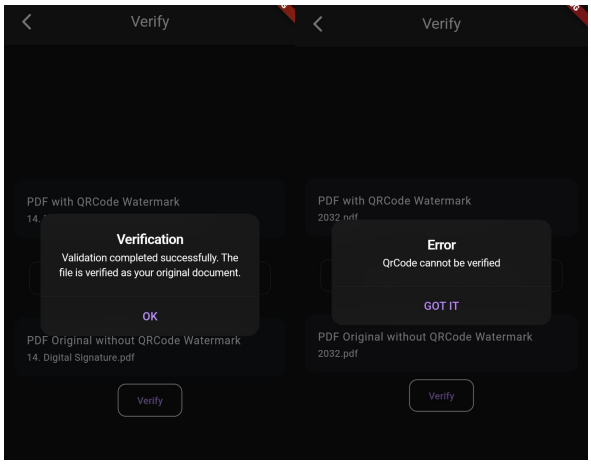


Fig. 17. Verification successful and Failed

The result obtained from verifying a PDF file that does not have a signature is that the verification fails because there is no QRCode in the file so that the application cannot continue the decryption process, and an error warning appears that says "QRCode cannot be verified" which means the QRCode cannot be verified. The scenario still uses a negative scenario, namely entering two different PDF files, which means that the original PDF file and the PDF file that has a QRCode are files that have different contents even though both have a QRCode. The next attempt showed a failed result because the two PDF files had different content so that their authenticity could not be checked because the checking process was measured based on bytes from the original PDF. From the results, a statement appeared that said "Validation failed, the file cannot be verified as your original document" which means validation failed and the file cannot be verified as an original document. The following are other test results listed in the table below:

TABLE I. Application Testing

Test Type	Description	Expected Result	Status
Functional Testing	PDF Signing	PDF successfully signed and signature verified correctly	*Passed*
	Signature Verification	The application displays the signature verification results correctly.	*Passed*
Security Testing	Private Key Storage	Private key cannot be extracted from device	*Passed*

V. Conclusion

The use of Digital Signatures with ED25519 and Asymmetric Encryption is a critical application of modern cryptographic technology. ED25519 provides a high level of security, efficiency, and ease of use, making it an ideal choice for digital signature processes. This method significantly enhances data security and simplifies key management,

addressing the inherent vulnerabilities of traditional Asymmetric Encryption systems. By integrating ED25519-based Digital Signatures into the Digitograph application, it ensures that PDF files can be securely signed, verified, and protected from unauthorized access or tampering. Digitograph not only guarantees the authenticity and integrity of documents but also offers a practical and efficient solution for secure interactions in an increasingly mobile and interconnected digital world. Additionally, the use of advanced cryptographic techniques in the application strengthens trust in digital transactions, making it an essential tool for individuals and organizations alike in safeguarding sensitive data.

REFERENCES

[1] N. Josias Gbetoho Saho and E. C. Ezin, 'Securing Document by Digital Signature through RSA and Elliptic Curve Cryptosystems', in *2019 International Conference on Smart Applications, Communications and Networking (SmartNets)*, 2019, pp. 1–6. DOI: 10.1109/SmartNets48225.2019.9069749

[2] S. Bugert, J. Heeger, and W. Berchtold, 'Integrity and authenticity verification of printed documents by smartphones', *Electronic Imaging*, vol. 35, no. 3, pp. 352--1-352–5, Jan. 2023. DOI: 10.2352/EI.2023.35.3.MOBMU-352

[3] F. Liu *et al.*, 'A survey on lattice-based digital signature', *Cybersecurity*, vol. 7, no. 1, p. 7, Apr. 2024. DOI: 10.1186/s42400-023-00198-1

[4] A. S. Kristiawan, F. R. Sanjaya, and F. H. Prasetya, 'Development of Blockchain-Based Digital Signature Platform', *Journal of Business and Technology*, vol. 2, no. 3, pp. 108–118, Dec. 2022. DOI: 10.24167/jbt.v2i3.4417

[5] K. Algazy, K. Sakan, A. Khompysh, and D. Dyusenbayev, 'Development of a New Post-Quantum Digital Signature Algorithm: Syrga-1', *Computers*, vol. 13, no. 1, p. 26, Jan. 2024. DOI: 10.3390/computers13010026

[6] J. Brendel, C. Cremers, D. Jackson, and M. Zhao, 'The Provable Security of Ed25519: Theory and Practice', in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 1659–1676. DOI: 10.1109/SP40001.2021.00042

[7] P. D. P. Adi, 'Security Performance of LoRaWAN Servers using Advanced Encryption Standard', *Internet of Things and Artificial Intelligence Journal*, vol. 3, no. 3, pp. 220–238, Aug. 2023. DOI: 10.31763/iota.v3i3.632

[8] K. Raut, 'A Comprehensive Review of Cryptographic Algorithms', *International Journal for Research in Applied Science and Engineering Technology*, vol. 9, no. 12, pp. 1750–1756, Dec. 2021. DOI: 10.22214/ijraset.2021.39581

[9] A. Fauzan, P. Sukarno, and A. A. Wardana, 'Overhead Analysis of the Use of Digital Signature in MQTT Protocol for Constrained Device in the Internet of Things System', in *2020 3rd International Conference on Computer and Informatics Engineering (IC2IE)*, 2020, pp. 415–420. DOI: 10.1109/IC2IE50715.2020.9274651

[10] H. K. Albahadily, I. A. Jabbar, A. A. Altaay, and X. Ren, 'Issuing Digital Signatures for Integrity and Authentication of Digital Documents', *Al-Mustansiriyah Journal of Science*, vol. 34, no. 3, pp. 50–55, Sep. 2023. DOI: 10.23851/mjs.v34i3.1278

[11] M. DAS and Z. Wang, 'ED25519: A New Secure Compatible Elliptic Curve for Mobile Wireless Network Security', *Jordanian Journal of Computers and Information Technology*, no. 0, p. 1, 2022. DOI: 10.5455/jjcit.71-1636268309

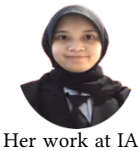
[12] A. Faz-Hernández, J. López, and R. Dahab, 'High-performance Implementation of Elliptic Curve Cryptography Using Vector Instructions', *ACM Transactions on Mathematical Software*, vol. 45, no. 3, pp. 1–35, Sep. 2019. DOI: 10.1145/3309759

[13] N. Khan, N. Sakib, I. Jerin, S. Quader, and A. Chakrabarty, 'Performance analysis of security algorithms for IoT devices', in *2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, 2017, pp. 130–133. DOI: 10.1109/R10-HTC.2017.8288923

[14] M. Bisheh-Niasar, R. Azarderakhsh, and M. Mozaffari-Kermani, 'Cryptographic Accelerators for Digital Signature Based on Ed25519', *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 7, pp. 1297–1305, Jul. 2021. DOI: 10.1109/TVLSI.2021.3077885

[15] M. Abd Zaid and S. Hassan, 'Survey on Modern Cryptography', *Journal of Kufa for Mathematics and Computer*, vol. 7, no. 1, pp. 1–8, Sep. 2021. DOI: 10.31642/JoKMC/2018/070101

[16] N. Josias Gbetoho Saho and E. C. Ezin, 'Securing Document by Digital Signature through RSA and Elliptic Curve Cryptosystems', in *2019 International Conference on Smart Applications, Communications and Networking (SmartNets)*, 2019, pp. 1–6. DOI: 10.1109/SmartNets48225.2019.9069749



Annisa Putri Wahyuni

Currently a researcher at the Institute of Advanced Informatics and Computing (IAICO) in Tasikmalaya, Indonesia, she specializes in the intersection of information systems, information technology, and sentiment analysis. Her work at IAICO combines rigorous theoretical exploration with practical implementation, particularly in the development and optimization of advanced information systems aimed at enhancing data processing capabilities and enabling more effective decision-making processes. In addition, she collaborates with multidisciplinary teams to explore innovative approaches for integrating emerging technologies, such as artificial intelligence, to further improve system efficiency and adaptability.

Arif Bramantoro

Currently a senior assistant professor in the School of Computing and Informatics, Universiti Teknologi Brunei, Brunei Darussalam. Previously, he was an Associate Professor in the Information Systems Department, Faculty of Computing and Information Technology in Rabigh, King Abdulaziz University, Saudi Arabia. From 2011 to 2016, he was an Assistant Professor in the Information Systems Department, the College of Computer Sciences and Information, Al-Imam Mohammad Ibn Saud Islamic University, Riyadh, Saudi Arabia. He was an expert researcher at the Information Services Platform laboratory, National Institute of Information and Communication Technology, Japan, from 2011 to 2012. He received Ph.D. from the Department of Social Informatics, Kyoto University, Japan, in 2011. He holds master's degree from the Faculty of Information Technology, Monash University, Melbourne, Australia in 2006. His bachelor's degree was obtained from the Department of Informatics, Institute Technology of Bandung, Indonesia in 2001. His research interests include service systems, business process workflow, and business intelligence. He is the author of more than 60 articles. He was a recipient of first quartiles congratulatory publications letter from UTB in 2023, the Research Excellence award in 2016 from the Deanship of Scientific Research, Al-Imam University, Saudi Arabia; and the Best Paper award from the IEEE International Conference on Cloud Computing in 2015.

Randi Rizal

He was born in Tasikmalaya, West Java, Indonesia. He is currently completing his PhD at the Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka (UTeM), Malaysia. He works as a lecturer

at the Department of Informatics, Faculty of Engineering, Siliwangi University, Indonesia. His current research interests include cybersecurity, digital forensics, and artificial intelligence. He has published numerous research papers in various national and international journals. Additionally, he serves as the Managing Editor of the journal Innovation in Research of Informatics (INNOVATICS) and as the Editor in Chief of the International Journal of Informatics and Computing (JICO) under the publisher Institute of Advanced Informatics and Computing, Indonesia. He actively participates in academic conferences, contributing to the dissemination of knowledge and the advancement of his fields of expertise. His dedication to research and education has earned him recognition within the academic community.
.

Souhayla Elmeftahi

Currently a researcher in the Data Engineering program at the National School of Applied Science, Morocco, she is passionate about leveraging advanced technologies to solve real-world problems. Her research interests encompass data engineering, data science, data analytics, Internet of Things (IoT), cloud computing, big data, machine learning, and related informatics fields. She aims to contribute to the development of innovative solutions and applications that enhance decision-making processes, optimize systems, and improve overall efficiency across industries. Additionally, she is interested in exploring the intersection of emerging technologies and societal impact.