INTERNATIONAL JOURNAL OF INTEGRATED ENGINEERING
ISSN: 2229-838X e-ISSN: 2600-7916

IJIE

Vol. 17 No. 9 (2025) 12-27
https://publisher.uthm.edu.my/ojs/index.php/ijie

# Enhancing Security Using ECDSA-Based Hardware Security Module with DS28E38 Secure Data Authentication IC

**Jazi Eko Istiyanto[1]*, Oskar Natan[1], Nia Gella Augoestien[1], Irfansyah Peradeniya[1], Syauqi Giffari Rachman[1], Mellia Putri[1], Joshua Tito Amael[1], Rifda Hakima Sari[1]**

[1] *Universitas Gadjah Mada, Department of Computer Science and Electronics, Bulaksumur, Depok, Sleman Regency, Special Region of Yogyakarta 55281, INDONESIA*

*Corresponding Author: jazi@ugm.ac.id
DOI: https://doi.org/10.30880/ijie.2025.17.09.002

**Article Info**

Received: 27 June 2025
Accepted: 12 December 2025
Available online: 31 December 2025

**Keywords**

IC Security, ECDSA, Authentication, Cyberattacks

**Abstract**

The increasing integration of technology across diverse sectors such as manufacturing, logistics, healthcare, and smart infrastructure has enhanced operational efficiency and introduced significant cybersecurity vulnerabilities due to open networks and automatic technology. This research addresses these risks by developing a verification system using the integrated circuit (IC) security DS28E38, which employs Elliptic Curve Cryptography (ECC) to keep data integrity. The key innovation lies in combining physically unclonable function (PUF)-based authentication with the Elliptic Curve Digital Signature Algorithm (ECDSA), providing a resilient mechanism for device identity verification and protection against cyber threats. The research also includes a comprehensive performance evaluation, considering metrics such as time efficiency, memory utilization, power consumption, and resilience against simulated cyberattacks, demonstrating that the proposed system significantly enhances cybersecurity while preserving operational performance.

## 1. Introduction

The industrial sector has undergone a significant transformation in recent years, driven by the integration of intelligent systems and automation technology. From manufacturing to logistics and healthcare, organizations are increasingly deploying connected devices and embedded hardware to streamline operations, reduce labor dependency, and enhance decision-making capabilities [1],[2]. This transition toward highly automated environments relies heavily on machine-to-machine (M2M) communication, real-time data exchange, and decentralized control systems, which are embedded in hardware [3],[4]. The increased connectivity between devices, controllers, and networks creates new entry points for cyber threats, especially when communication protocols and hardware interfaces lack built-in security mechanisms [5]. Alarmingly, recent reports show a growing number of anomalies and cyber incidents in industrial communication systems, with millions of suspicious events recorded annually [4], [5].

Cybersecurity in industries faces significant challenges because most of these systems are connected to open networks that are accessible remotely. Research shows that attacks such as man-in-the-middle (MITM), ransomware, and Denial-of-Service (DoS) are becoming more frequent [6], [7]. Beyond technical controls, recent work published in this journal emphasizes that effective cybersecurity also depends on organizational and human factors, including security culture and policy compliance among both IT and non-IT personnel [35].

For instance, in attacks involving machine control systems, attackers can manipulate commands or steal sensitive data transmitted from the machines to a central server, which can lead to operational failure or significant losses in production efficiency [8].

The impact of these cyberattacks can be severe, mainly when industrial machines rely on the reliability of real-time data. Recent research indicates that attackers can exploit weaknesses in networks and control systems, as demonstrated in the infamous Stuxnet incident, where industrial systems were compromised by injecting malicious code into automated control systems [9]. These trends highlight the need for hardware-level security strategies to ensure trusted authentication, prevent data tampering, and defend against a wide range of evolving cyber threats.

This research aims to develop a robust authentication system for industrial machines that need security protection by utilizing the DS28E38 IC, which integrates the Elliptic Curve Digital Signature Algorithm (ECDSA) based cryptographic solutions. The study enhances communication security between general industrial machines and central servers by addressing vulnerabilities such as replay attacks, data forgery, and unauthorized access [10]. By implementing DS28E38 ICs, the research proposes a scalable, hardware-based security framework designed to significantly reduce the risk of cyberattacks in industrial general machines such as Internet of Things (IoT) Systems in industry, Autonomous Mobile Robot, and Scada Systems, ensuring data integrity and authentication for reliable autonomous operations. The results of this study could serve as a benchmark for improving industrial machine security across various industrial applications.

The novelties of our research include:

- This research introduces a novel approach by integrating Physical Unclonable Function (PUF)-based authentication with the ECDSA in a system specifically designed for implementation on hardware devices.
- The research offers a new method of utilizing the DS28E38 integrated circuit (IC) to verify device identities.

Fig. 1 shows the overall system overview. This paper is organized as follows: In Section II, we conduct a comprehensive review of several related studies that inspired this research. Section III presents the proposed model, mainly focusing on the verification system and system evaluation. Section IV analyzes the research results to evaluate time, memory, power consumption, and attack simulations. Finally, in Section V, we conclude the findings of the research.

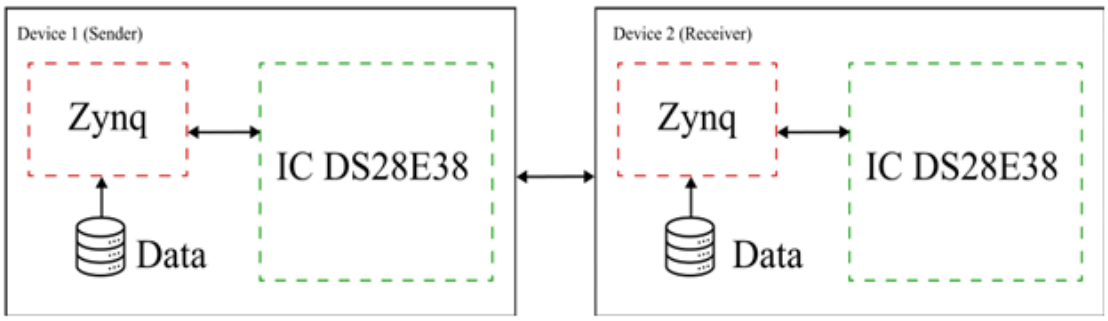


**Fig. 1** *System overview*

The diagram shows a system involving two devices, each with a Zynq processor connected to a DS28E38 IC for secure data exchange. The DS28E38 IC implements an ECDSA and a PUF to ensure trusted authentication, prevent data tampering, and secure communication. This setup addresses cybersecurity challenges in industrial environments, protecting against attacks like man-in-the-middle, ransomware, and unauthorized access, ensuring the integrity of real-time data and the reliability of hardware security module operations.

## 2. Related Works

This section reviews existing research on cryptography, focusing on ECDSA, authentication mechanisms, and integrated circuit security. The key concepts from these studies are then identified to provide inspiration for the current work and establish a framework for comparative analysis.

Int. Journal of Integrated Engineering Vol. 17 No. 9 (2025) p. 12-27

## 2.1 Cryptographic ECDSA

Cryptographic algorithms, particularly the ECDSA, have emerged as one of the most effective solutions for ensuring data integrity and security on resource-constrained devices [11]. The effectiveness of ECDSA stems from its unique combination of high security, computational efficiency, and scalability, which are critical characteristics of modern cryptographic systems.

One of the most significant advantages of ECDSA is its ability to provide security levels equivalent to traditional algorithms like RSA but with significantly smaller key sizes. For instance, a 256-bit ECDSA key offers comparable security to a 3072-bit RSA key, reducing the computational overhead and memory requirements [12]. This characteristic makes ECDSA particularly suitable for environments with limited resources, such as embedded systems and IoT devices. For example, Abidi et al. demonstrated the implementation of ECDSA on Field Programmable Gate Array (FPGA) devices, showing how a hardware-based approach can optimize performance by reducing computational load, which is crucial for real-time applications in embedded systems and IoT networks [12]. This highlights the importance of optimizing ECDSA's performance, especially in environments where computational resources are limited.

Recent advancements have further improved the efficiency of ECDSA implementations. Studies have shown that reducing the number of elliptic curve point operations can enhance system performance, making it suitable for applications requiring fast responses and low power consumption, such as embedded systems or devices with limited computational capabilities [14]. Additionally, integrating ECDSA with advanced hash functions, such as the Pizer hash function, has proven effective in mitigating collision attacks while maintaining computational simplicity [15]. These innovations are particularly relevant given the increasing threats to data integrity across various sectors, including cybersecurity, finance, and healthcare.

Furthermore, comparative analyses of digital signature schemes have demonstrated ECDSA's versatility. Modified ECDSA variants and hybrid encryption methods offer a flexible framework for balancing efficiency and security [16]. For instance, hybrid approaches that integrate ECDSA with symmetric encryption techniques have been successfully applied in secure communication protocols for industrial automation. This underscores the adaptability of ECDSA as a robust platform for addressing diverse digital security challenges.

## 2.2 Authentication

Authentication plays a central role in maintaining the validity of identities and preventing unauthorized access, especially in an era where the number of devices connected to public networks is rapidly increasing. ECDSA has emerged as a reliable solution for authentication, particularly in schemes that require high security and performance efficiency [17]. For example, implementing ECDSA in the BeiDou-II satellite navigation system successfully offers an additional security layer that prevents spoofing attacks, ensuring that users' navigation messages are authentic and reliable [18][19]. The importance of this authentication scheme is even more pronounced in applications involving cross-network communication, such as transportation and defense, where authentication errors could have fatal consequences.

Moreover, the ECDSA certificate-based authentication scheme in Advanced Metering Infrastructure (AMI) highlights how ECDSA can be applied to secure intelligent grid networks. By protecting the data exchange between smart meters and utility servers, this scheme not only enhances network security but also ensures the reliability of energy measurements in increasingly digitally connected systems [20]. Furthermore, studies on authentication schemes in Vehicular Ad Hoc Network (VANET) have also demonstrated the effectiveness of ECDSA in safeguarding communication between vehicles from cyberattacks that could compromise driver safety [21].

The challenge-response asynchronous authentication approach using smart cards in cloud environments provides a practical solution for flexible and secure authentication in a world increasingly reliant on cloud computing [13]. This approach's flexibility and security suit dynamic environments well, where devices with varying authentication levels operate within an integrated system.

## 2.3 Integrated Circuit

Hardware security has increasingly become a focal point in the modern era, where hardware often serves as an entry point for security threats if not properly protected. One promising approach is using Hardware Security Modules (HSM) in systems with complex integration. For instance, in the context of system-in-package (SiP), root-of-trust modules have been proven capable of providing solid protection in the increasingly heterogeneous chip integration process [22]. These modules ensure that processed data remains secure, even in environments rife with cyber threats.

In addition, hardware-based approaches are also being applied in the healthcare sector, as revealed by research on the security of ventilators. Integrating HSM with SoftHSM for medical devices like ventilators aims to protect sensitive patient data and ensure that medical devices comply with the highest security standards [6]. This

is particularly relevant in the modern era, where connected medical devices are increasingly targeted by cyberattacks, making hardware-based security solutions even more urgent.

Blockchain technology is also beginning to be integrated with hardware security modules, as seen in the SaFe framework for protecting smart vehicles. The combination of security elements and blockchain creates an ecosystem resistant to cyberattacks, especially in the automotive sector, which increasingly relies on vehicle-to-vehicle communication [23], [24], [25]. At the same time, cryptography-based HSM approaches in cloud environments also offer additional protection in managing cryptographic keys and data, which is essential for safeguarding sensitive data in cloud computing [26]. This is further reinforced by research on enhanced security schemes for mobile devices using hardware cryptographic modules, which provide an extra layer of protection against potential attacks on devices frequently accessed from different networks [27]. Similarly, in this research, we developed a digital signature verification technique using the IC DS28E38.

Similarly, in this research, we developed a digital signature verification technique using the IC DS28E38. This work aligns with the scope of IJIE, which has published on secure embedded systems, such as the use of a Zynq SoC for secured wireless image transmission [33] and performance evaluations of processor cores on FPGA platforms [34]. Furthermore, the journal has highlighted the importance of security beyond pure technology, including human and organizational factors [35]. Our ECDSA-based HSM contributes to this discourse by providing a hardware-rooted authentication mechanism that enhances technical security, thereby supporting a more robust overall security posture for industrial systems.

## 3. Implementation

### 3.1 Proposed Model

The proposed architecture for securing Inter Device Communication using a HSM leverages a combination of FPGA Zynq system-on-chip (SoC) and an external IC, namely the DS28E38, to provide an authentication mechanism. As shown in Fig. 2, Device 1 is the sender, while Device 2 is the receiver. This architecture utilizes UDP communication between the two devices and integrates ECDSA cryptographic elements to ensure that the data transmitted from Device 1 authentically originates from it and remains unaltered by any third party. The main components of Device 1 (the sender) include the Zynq SoC, which generates and hashes the data to be transmitted. Then, the IC DS28E38 is used to sign this hash using the ECDSA.

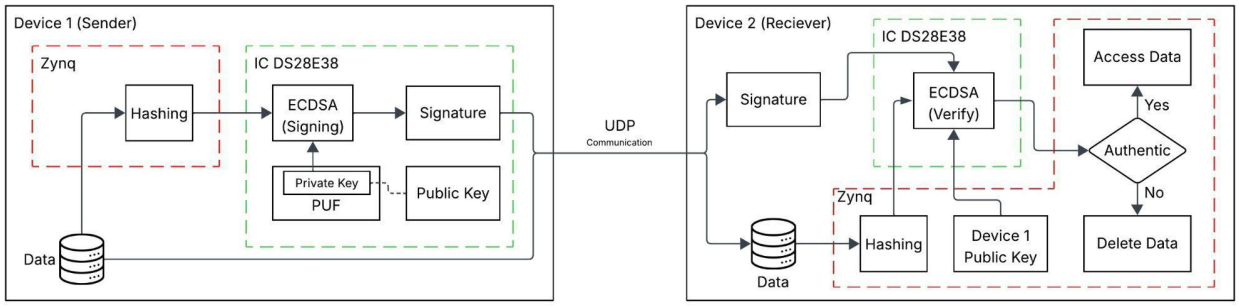


**Fig. 2** *System architecture*

The core processing unit in this architecture is the FPGA Zynq Ultrascale+ MPSOC, chosen for its advantages in hash computation and inherent security features. This FPGA enables design flexibility and security updates without hardware changes. Its hybrid ARM-FPGA architecture has been successfully leveraged in prior IJIE research for real-time embedded security applications, such as secured wireless image transmission in wildlife surveillance systems using SoC FPGA platforms[33]. Features such as large logic cells, integrated memory, and hardware acceleration support complex cryptographic processes and hardware-based authentication, making it an ideal platform for integrating a Hardware Security Module (HSM) in resource-constrained security applications.

The IC DS28E38 in this architecture plays a crucial role as the signature generator for ECDSA, utilizing the SHA256 hash input and a private key. The main advantage of this IC is that the private key is derived from the physical characteristics of the silicon wafer, a function known as a PUF. The PUF in the IC generates a unique cryptographic key that cannot be replicated, thus enhancing system security by authenticating legitimate devices [28]. By leveraging the physical variations of hardware materials, the PUF ensures that the original device can only use the private key, reducing the risk of cryptographic attacks.

The authentication process in the HSM architecture can be divided into two main stages: signature generation (e.g., ECDSA Signature) and signature verification using a public key. The first stage, signature generation, begins

with the collection of data to be transmitted. This data is then processed through a hashing algorithm, which generates the SHA-256 digest. This digest is a cryptographic representation of the original data and forms the basis for creating the ECDSA signature. Once the SHA-256 digest is created, the ECDSA algorithm generates a digital signature. At the end of this stage, the resulting signature is combined with the original data, forming a secure data package.

Similarly, in this research, we developed a digital signature verification technique using the IC DS28E38. This work aligns with the scope of IJIE, which has published on secure embedded systems, such as the use of a Zynq SoC for secured wireless image transmission [33] and performance evaluations of processor cores on FPGA platforms [34]. Furthermore, the journal has highlighted the importance of security beyond pure technology, including human and organizational factors [35]. Our ECDSA-based HSM contributes to this discourse by providing a hardware-rooted authentication mechanism that enhances technical security, thereby supporting a more robust overall security posture for industrial systems..

## 3.2 IC DS28E38 Secure Authenticator

The internal architecture of the DS28E38 is designed to provide high-security public key authentication through several key components, as shown in Fig. 3. The 1-Wire Interface (e.g., 1-Wire INFC and CMD) is an efficient communication interface, requiring only a single pin for data communication. The ChipDNA technology generates a unique private key based on the physical variations of the hardware, which cannot be cloned. It is used to secure data and perform digital signature operations using the ECC-P256 algorithm. The ECC-P256 Engine supports elliptic curve cryptography following the NIST P-256 standard, enabling the creation and verification of digital signatures using ECDSA. The true random number generator (TRNG) ensures the security of cryptographic processes by generating random numbers that meet FIPS/NIST standards. At the same time, the 64-bit ROM ID provides a unique identifier for the device. The 2Kb EEPROM stores cryptographic keys and certificates with cryptographic protection, and the Decrement Counter allows control over the number of devices used. Support for parasitic power operations through the CEXT pin allows the device to harvest energy from external sources, further enhancing energy efficiency. This feature makes the architecture well-suited for low-power applications like battery-operated devices or IoT systems.

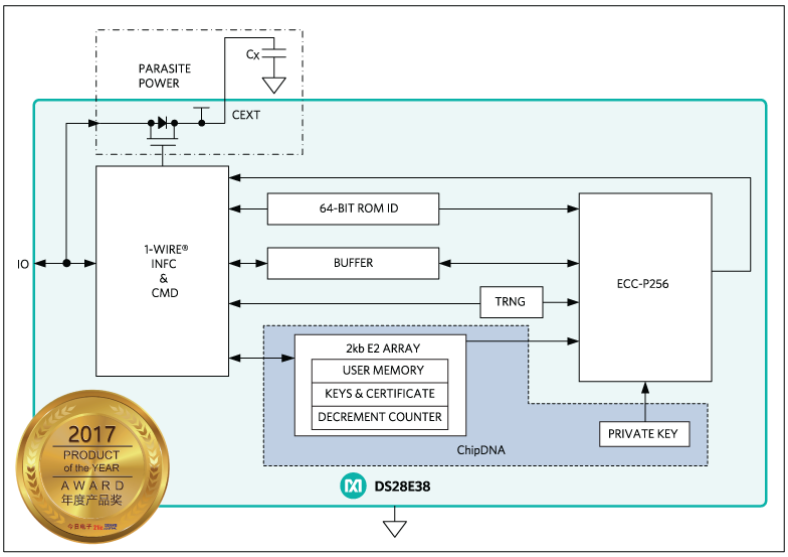


*Fig 3. IC Architecture*

The process begins by collecting several parameters, including the ROMID, page data, challenge, and MANID. These parameters are then combined and processed through the SHA-2 algorithm to generate a message digest or hash of the data, denoted as $h(m)$. As illustrated in Fig. 4, the ECDSA signature algorithm uses two main inputs: a random value $k$, generated by a Random Number Generator (RNG), and a private key $d$, which can be derived from a PUF or a specific page of data stored on the device.

Combining the random value $k$, the private key $d$, and the message digest $h(m)$, the algorithm produces a pair of r and s values constituting the ECDSA signature's components. The final output is a signature represented as $[r, s]$, which is then sent along with the original data to be validated by the receiver.

The signature verification process begins when the receiver retrieves the original data, which includes the ROMID, page data, challenge, page number, and MANID—the same parameters used during the signature creation step. This data is processed again using the SHA-2 algorithm to regenerate the message digest $h(m)$.

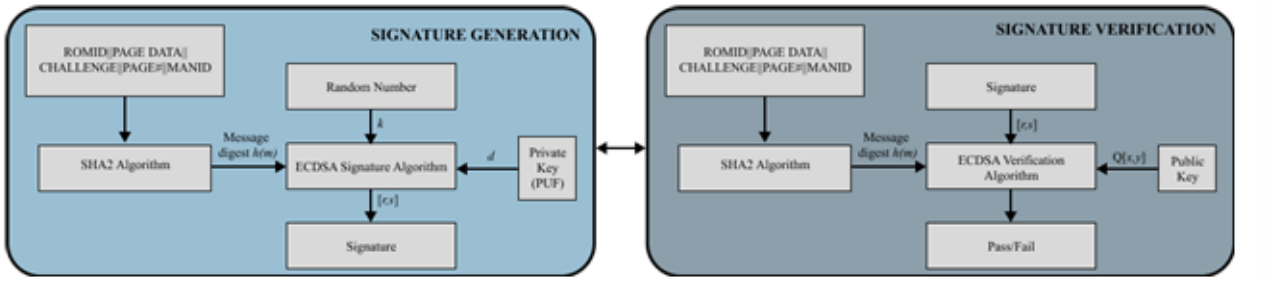


**Fig. 4** *ECDSA workflow (Signature generation - verification)*

The received signature $[r, s]$ is then input into the ECDSA verification algorithm and the sender's public key, denoted as $Q[x, y]$. This process aims to verify whether the signature matches the regenerated message digest. If the verification result matches, the data is considered valid, and the process concludes with a "pass" status. Conversely, if the verification fails, the result is marked as "fail," indicating that the signature is invalid or the data has been altered.

## 3.3 Verification

The verification process is carried out through several steps as follows:

## ECDSA Signature Generation

1. **Step 1: Hashing Data**

   The message m is hashed using the SHA-2 algorithm to produce a fixed-length message digest $h(m)$. Hashing transforms the message into a fixed-size output (e.g., usually 256 bits for SHA-256), which serves as the input for the signature algorithm.

$$h(m) = SHA - 2(m) \tag{1}$$

2. **Step 2: Generating Random Value $k$**

   A cryptographically secure random value $k$ is selected from the interval $[1, n-1]$, where $n$ is the order of the elliptic curve. The random value $k$ is crucial because using the same $k$ for multiple signatures compromises security.

$$k \in [1, n-1] \tag{2}$$

3. **Step 3: Calculating $r$**

   The scalar multiplication of the random value $k$ with the elliptic curve's generator points $G$ produces a point on the elliptic curve. The x-coordinate of this point is taken modulo $n$, and this value becomes $r$, the first component of the ECDSA signature.

$$r = (k \cdot G)_x mod_n \tag{3}$$

   Here, $G$ is the generator point, and $(k \cdot G)_x$ is the x-coordinate of the resulting point on the elliptic curve. If $r = 0$, the process is repeated with a different $k$.

4. **Step 4: Calculating $s$**

   The second part of the signature $s$ is computed using the private key $d$, the message $h(m)$ hash, and the previously calculated $r$. The formula for $s$ is:

$$s = k^{-1} \cdot (h(m) + d \cdot r) mod_n \quad (4)$$

**Where:**

$d$ is the private key.

$h(m)$ is the hash of the message (from Step 1).

$k^{-1}$is the modular inverse of □ *mod* □ (from Step 2).

If $s = 0$, the process is repeated with a new random □.

5. **Signature Output**

The generated signature is the pair $[r, s]$, which is sent along with the original data.

$$Signature = [r, s] \quad (5)$$

The pair $[r, s]$(Eq. 5) represents the unique signature for the message, ensuring that anyone with the sender's public key can verify its authenticity. The algorithm used in this phase is outlined below as Algorithm 1, which provides a detailed step-by-step procedure for generating the ECDSA signature. The verification system follows the procedure outlined in Fig. 4.

The ECDSA workflow verifies a message's authenticity and integrity using public key cryptography. The signature generation uses a private key and a random number, while the verification process checks the signature against the device's public key to ensure it matches the data. This ensures secure communication between the sender and receiver.

## ECDSA Signature Verification

1. **Step 1: Hashing the Received Data**

The received message m is hashed again using the same SHA-2 algorithm to produce the message digest $h(m)$, which is used for verification. This ensures that the received message matches the one that was signed.

$$h(m) = SHA - 2(m) \quad (6)$$

2. **Step 2: Calculating the Modular Inverse of $s$**

The recipient calculates the modular inverse of *s* mod *n*. This step is crucial for recovering the original random value used during signature generation.

$$S^{-1} mod_n \quad (7)$$

3. **Step 3: Calculating $u_1$ and $u_2$**

Using the modular inverse of $s$ (Eq. 7), the recipient calculates two intermediary values, $u_1$ and $u_2$. These values are used to compute a point on the elliptic curve.

$$u_1 = h(m) \cdot S^{-1} mod_n \quad (8)$$

$$u_2 = r \cdot S^{-1} mod_n \quad (9)$$

Here:

$u_1$ is derived from the message hash (Eq. 6).

$u_2$ is derived from the first part of the signature $r$ (Eq. 3).

4. **Step 4: Calculating the Point $(x, y)$**

The recipient computes a point on the elliptic curve using the values $u_1$ (Eq. 8) and $u_2$ (Eq. 9), the generator point $G$, and the sender's public key $Q$.

$$(x, y) = u_1 \cdot G + u_2 \cdot Q \quad (10)$$

Where:

$G$ is the generator point.

$Q$ is the sender's public key.

$(x, y)$ is the resulting point on the elliptic curve.

**5. Step 5: Verifying $r$**

The signature is considered valid if the x-coordinate (Eq. 10) of the calculated point matches the value $r$, modulo $n$. Suppose the computed $r$ equals the value $r$ from the signature, which is valid.

$$r = xmod_n \tag{11}$$

If the verification fails, it indicates that the signature does not match the data or that the sender's public key is incorrect. The algorithm used in this phase is as follows: Algorithm 2.

**Algorithm 1: ECDSA Signature Generation**
**Input**: Message *m*, private key *d*, elliptic curve parameters (*G*, *n*)
**Output**: Signature (*r, s*)
Compute the message hash $h$ = SHA-2($m$)
Select a random value $k \in [1, n-1]$
Compute $r = (k \cdot G)x \bmod n$
**if** $r = 0$ **then**
Repeat the process with a new random $k$
**end if**
Compute $s = k^{-1} (h + d \cdot r) \bmod n$
**if** $s = 0$ **then**
Repeat the process with a new random $k$
**end if**
The signature is the pair ($r$, $s$)
return ($r$, $s$)

**Algorithm 2**: ECDSA Signature Verification
**Input**: Received message *m*, signature (*r, s*), public key *Q*, elliptic curve parameters (*G, n*)
**Output**: Valid / Invalid signature
Compute the message hash $h$ = SHA-2($m$)
Compute the modular inverse $s^{-1} \bmod n$
Compute $u1 = h \cdot s^{-1} \bmod n$
Compute $u2 = r \cdot s^{-1} \bmod n$
Calculate the elliptic curve point: $(x, y) = u1 \cdot G + u2 \cdot Q$
Verify that $r = x \bmod n$
**if** $r = x \bmod n$ **then**
The signature is valid
**else**
The signature is invalid
**end if**

Int. Journal of Integrated Engineering Vol. 17 No. 9 (2025) p. 12-27

### 3.4 System Evaluation

Aspects that can be evaluated from this HSM system include several key metrics that can be evaluated to assess its efficiency and security. Verification time is critical, as the duration required to verify the signature must be short enough not to interfere with the overall system performance [29], [30]. Memory and CPU usage during verification must also be analyzed, given that the verification process involves intensive calculations, such as hashing and elliptic curve operations, which can impact memory resource usage [31], [32]. In addition, attack simulations should be conducted to assess the system's resilience against various types of attacks, such as brute force attacks or attacks based on weak keys, to ensure the system's security against cryptographic threats. Finally, power consumption during verification is of particular concern, especially in applications on devices sensitive to energy consumption, such as autonomous robots. Low power consumption is essential for maintaining operational time without compromising security levels.

## 4. Result

In this section, we discuss the test results obtained from the fully implemented HSM system prototype. The physical hardware testbed, shown in Figure 5, was used for all experiments, including data integrity tests, attack simulations, and performance measurements for time, memory, and power consumption. This setup validates that the presented results reflect the system's real-world operational characteristics.

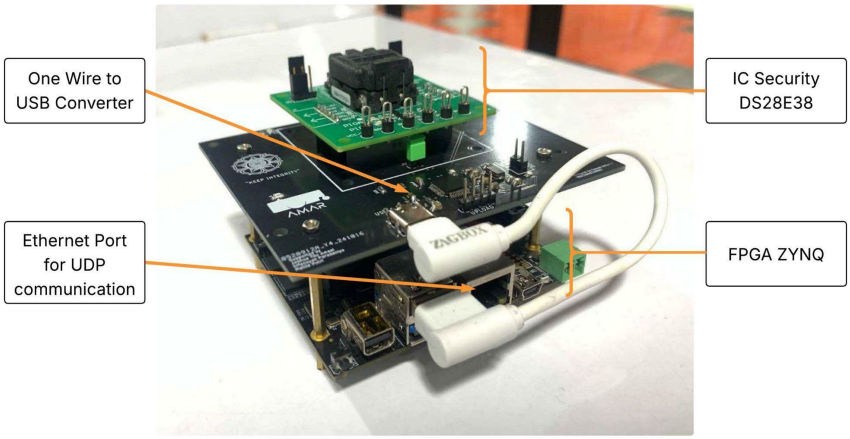


**Fig. 5** *Prototype hardware implementation of the proposed HSM system*

### 4.1 Data Integrity

For data to be considered secure, the HSM must ensure the integrity of the data while communicating through insecure channels. Therefore, this test includes a data integrity assurance process. The test involves running one cycle of data verification and reviewing whether the data remains the same as before processing or if any changes have occurred. The cycle begins with a 10 MB sample of data sensors and communication. This data consists of sensor log data sent from Device 1 to Device 2, as shown in Fig. 6. The process continues with creating a file signature using the DS28E38 IC Security. The process is carried out at this stage using the PUF within the IC Security. This process is illustrated in Fig. 7. The resulting signature is stored in the same memory as the original. The content of the generated signature is shown in Fig. 8.

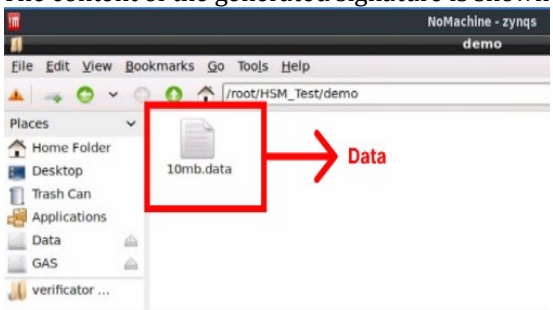


**Fig. 6** *Secured data*

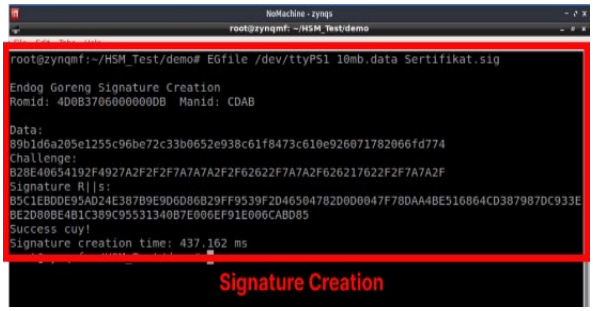


**Fig. 7** *Signature creation*

The data from Device 1, consisting of the signature and the original data, is combined into one file and transferred to Device 2, as shown in Fig. 9. The HSM module on Device 2 immediately verifies the incoming data. This verification process involves the ECDSA cryptographic algorithm to check whether the original data integrity is kept by comparing the existing signature with the received data. This process is illustrated in Fig. 10. If the data has not been altered, the HSM system provides the result shown in Fig. 11.


**Fig. 8** *Signature*

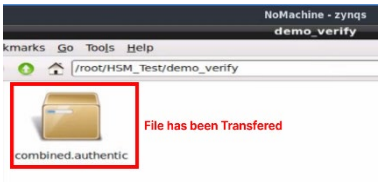


**Fig. 9** *Transferring file to device 2*

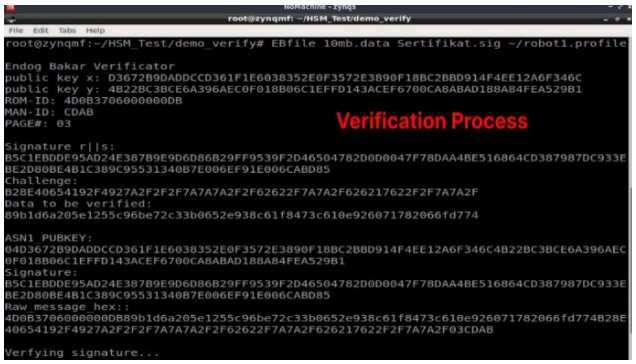


**Fig. 10** *Verification process*

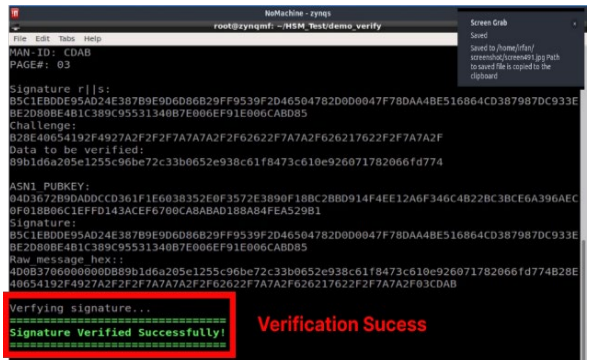


**Fig. 11** *Verification result*

## 4.2 Attacking Simulation

In this test, a simulated attack evaluates the system's response when a third party forcibly alters the data transmitted to Device 2. The test begins by modifying the original data, as shown in Fig. 12, into the altered version depicted in Fig. 13. At this stage; Device 2 performs a re-validation process to determine whether the received data remains authentic in terms of both content and sender identity.

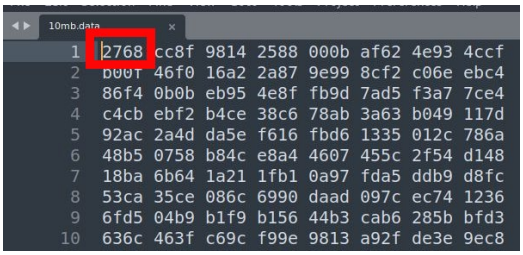


**Fig. 12** *Target before attacking*

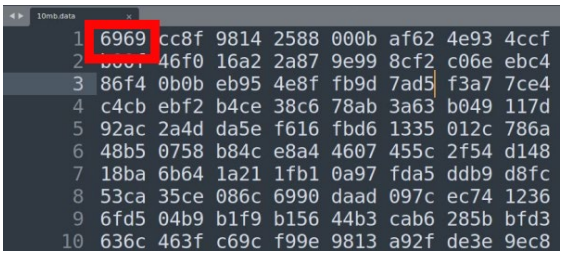


**Fig. 13** *Target after attacking*

This experiment found that the HSM system on Device 2 could detect that the data sent to it was not authentic and had been altered. The verification system of Device 2 successfully detected that the data had been altered, as indicated by the warning shown in Fig. 14.

Int. Journal of Integrated Engineering Vol. 17 No. 9 (2025) p. 12-27

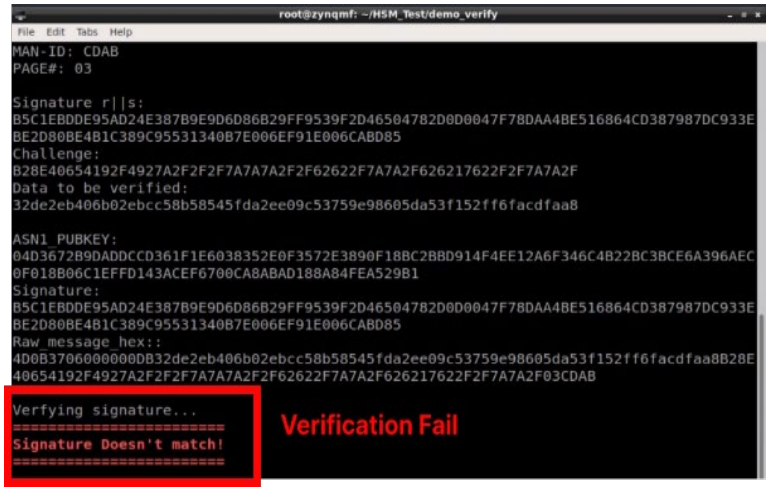


**Fig. 14** *Attacking result*

## 4.3 Time Consumption

A HSM has a key variable, one of which is the efficiency of time consumption. This is because an HSM greatly emphasizes data processing time [6]. Therefore, in this study, time consumption testing was conducted to measure the time required by the HSM when performing the signature creation process with the assistance of IC security on a file. The testing was conducted with varying file sizes ranging from 1 MB, 10 MB, 50 MB, 100 MB, and 500 MB. Three types of time consumption were measured: real-time, user time, and system time. Real-time is the total elapsed time from the start to the end of the program execution, user time is the time spent by the CPU executing instructions from the program in user space, and system time is the time spent by the CPU running the operating system functions on behalf of the program. The results of this test are the average values obtained from ten trials for each file size. The results of this test are shown in Table 1.

**Table 1** *Signature process file times*

| File Size (mb) | Real (s) | User (s) | System (s) |
|---|---|---|---|
| 1 | 0.485 | 0.30 | 0.200 |
| 10 | 0.667 | 0.203 | 0.024 |
| 50 | 1.493 | 0.997 | 0.026 |
| 100 | 2.487 | 1.903 | 0.108 |
| 500 | 22.708 | 9.556 | 0.632 |

Based on the data in Table 1, it is observed that processing time increases as the file size being processed grows. For a 1 MB file, the real-time is 0.485 seconds, with a user time of 0.30 seconds and a system time of 0.20 seconds. When the file size is increased to 10 MB, the time increases only slightly, with a real-time of 0.667 seconds, user time of 0.203 seconds, and system time of 0.024 seconds. However, for larger files, such as 500 MB, there is a significant jump in processing time, with real-time reaching 22.708 seconds, user time 9.556 seconds, and system time 0.632 seconds.

These results show that the HSM system efficiently handles small to medium-sized files with relatively short processing times. However, processing time significantly increases for larger files, particularly user time. This result indicates that although the HSM can handle large volumes of data, processing time becomes longer, which could be a limiting factor in situations that require real-time data processing.

## 4.4 Memory Consumption

In this test, the HSM's CPU memory and main memory usage were measured during data processing. The testing was conducted by calculating the average memory consumption required for signature creation processing from ten trials with different file types, as shown in the Table 2.

*Table 2 Memory consumption in signature creation*

| File Size (mb) | Memory (%) | CPU (%) |
|---|---|---|
| 1 | 0.0 | 12.7 |
| 10 | 0.0 | 15 |
| 50 | 0.0 | 21.2 |
| 100 | 0.0 | 33 |
| 500 | 0.0 | 47.2 |

Table 2 shows the measurement results of memory and CPU usage during the signature creation process on the HSM IC Security. CPU usage increases gradually as the file size being processed grows. For a 1 MB file, CPU usage is 12.7%, while for a 500 MB file, CPU usage rises to 47.2%. This indicates that the larger the file being processed, the greater the load placed on the CPU. This increase in CPU usage occurs because the cryptographic process performed by the HSM requires more processing resources to generate a signature for larger files.

Meanwhile, memory usage remains stable at 0% for all file sizes, from 1 MB to 500 MB. The HSM does not require significant main memory to process data during the signature creation. Perhaps the memory system used by the HSM is well-optimized, so it does not strain the main memory even as file size increases. Thus, it can be concluded that the signature creation process on the HSM IC Security heavily relies on CPU usage, especially when processing larger files, while memory usage remains minimal and efficient throughout the process.

## 4.5 Power Consumption

This test measures the voltage, current, and power the HSM module requires during a single signature creation process. The data is collected using five different file sizes. The calculations for this test are based on the average values obtained from ten trials for each file size.

Fig. 15, Fig. 16, and Fig. 17 present the measurement results of current (A), power (watt), and voltage (V) level during the signature creation process on the HSM IC Security for various file sizes. The measured voltage remains stable at 12.30 V across all file sizes, ranging from 1 MB to 500 MB, indicating that the HSM system operates at a constant voltage. On the other hand, the current used gradually increases with the growing file size. For a 1 MB file. The current is 0.617 A, rising to 0.645 A for a 500 MB file. This increase in current indicates that the larger the file being processed, the more current is required to support the signature creation process. As a result, power consumption (Watt) also increases, from 7.59 Watts for a 1 MB file to 7.93 Watts for a 500 MB file. This rise in power is consistent with the increase in current, as power is the product of voltage and current. Thus, even though the voltage remains constant, the increase in current leads to higher power consumption when the HSM processes larger files. In conclusion, this table shows that the signature creation process on the HSM requires more power and current as file sizes increase, even though the voltage remains stable.

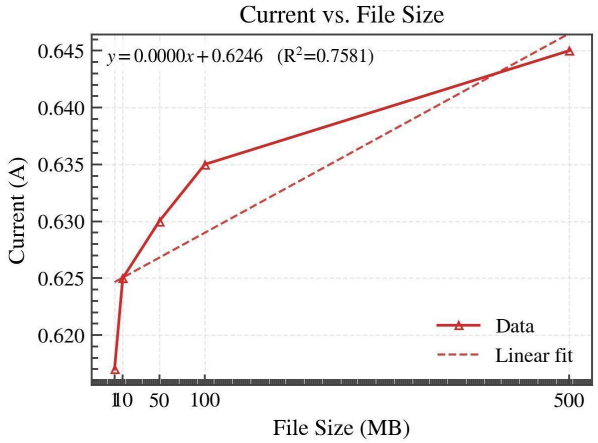


**Fig. 15** *Current consumption in signature creation*

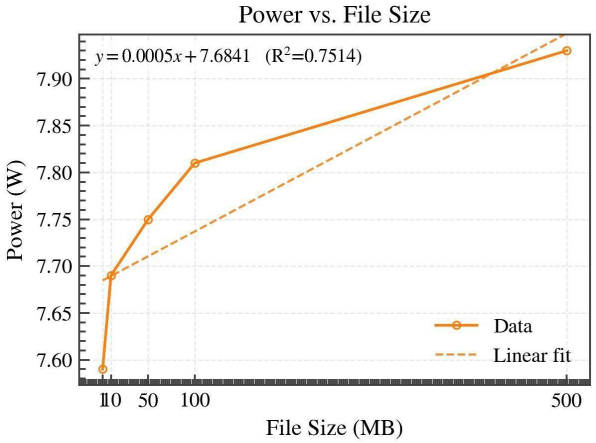


**Fig. 16** *Power consumption in signature creation*

Int. Journal of Integrated Engineering Vol. 17 No. 9 (2025) p. 12-27

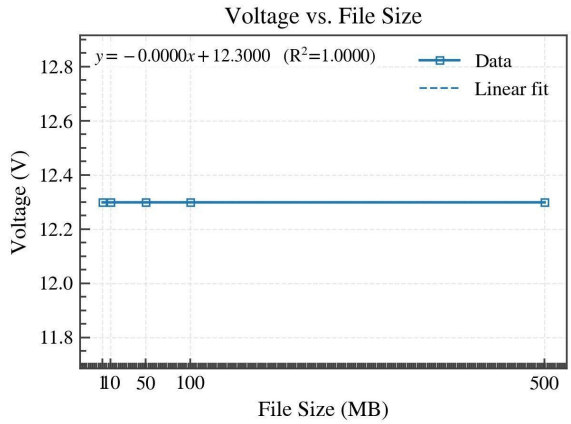


**Fig. 17** *Voltage level in signature creation*

## 5. Discussion

The experimental evaluation of the ECDSA-based HSM incorporating the DS28E38 secure authentication IC demonstrates its capability to deliver hardware-rooted data integrity protection in embedded systems. The data integrity and attack simulation tests (Fig. 6–Fig. 14) confirm that the architecture can reliably detect unauthorized data alterations, consistent with the expected performance of elliptic-curve-based authentication systems reported in prior studies. By leveraging the DS28E38's PUF-derived private key in combination with ECDSA, the system ensures that both message content and source identity are cryptographically bound, which is essential for resisting man-in-the-middle and replay attacks in unsecured networks.

From a performance perspective, the time consumption results (Table 1) reveal a near-linear increase in real-time execution and CPU utilization as file size increases, with minimal system time overhead. This suggests that signature creation is predominantly a user-space cryptographic workload, heavily dependent on processing throughput rather than memory bandwidth. The negligible main memory usage across all file sizes (Table 2) further supports the conclusion that the HSM's design is memory-efficient and well-suited for devices with limited RAM capacity. This aligns with findings by Chung et al. [34], who demonstrated that FPGA-based embedded systems can achieve high resource efficiency while maintaining computational performance through careful core and peripheral optimization. However, the rise in CPU load from 12.7% at 1 MB to 47.2% at 500 MB indicates that computational scaling may become a bottleneck for high-frequency or large-batch authentication tasks, particularly in real-time systems.

Power consumption measurements (Fig. 15–Fig. 17) show that while the supply voltage remains stable at 12.30 V, current draw and resulting power usage increase proportionally with file size. Although the increase is modest—from 7.59 W to 7.93 W—it reflects the direct correlation between processing effort and energy demand in hardware-assisted cryptographic operations. This finding aligns with prior observations that computation-intensive security functions on embedded hardware can influence both energy efficiency and thermal behavior.

These results collectively highlight several scientific and engineering implications. First, the combination of PUF-based key generation and elliptic curve cryptography in a hardware module provides a robust approach to securing communication in hostile environments without imposing significant memory overhead. Second, the predictable scaling behavior in both processing time and power suggests that system designers can estimate resource requirements with high accuracy during integration. Third, the performance trends indicate that optimizing CPU-bound cryptographic workloads—potentially through FPGA hardware acceleration on the Zynq SoC—could substantially improve throughput for large data sets while keeping power budgets within acceptable limits.

The architectural approach validated here is therefore relevant not only for industrial IoT and autonomous robotic systems but also for any embedded application requiring verifiable, low-latency data authentication under constrained resources. Extending the implementation to a full FPGA Zynq SoC environment with the DS28E38 IC enables the measurement of hardware-specific parameters such as logic utilization, interconnect latency, and on-chip integration overheads. Such a step would provide deeper insight into the trade-offs between cryptographic security strength, resource utilization, and real-time performance in embedded security modules.

## 6. Conclusion

Based on the test results, the ECDSA-based HSM utilizing the DS28E38 IC has significantly improved data security and maintained data integrity. The system effectively detected changes or inconsistencies in sender information during data transfer processes. CPU usage increased with growing file size, from 12.7% for a 1 MB file to 47.2%

for a 500 MB file, while memory usage remained consistently efficient at 0%. Power consumption also rose with file size, with current increasing from 0.617 A to 0.645 A and power usage from 7.59 watts to 7.93 watts, despite the voltage remaining stable at 12.30 V. Processing time significantly increased, from 0.485 seconds for a 1 MB file to 22.708 seconds for a 500 MB file. Overall, the HSM demonstrates exceptional efficiency in memory usage and has proven effective in securing data transfers. However, careful optimization strategies may be required to address the increasing CPU usage, current, power consumption, and processing time for larger files to ensure optimal performance and scalability. Additionally, the system's ability to detect data tampering and maintain consistency in sender identification highlights its potential as a robust solution for industrial cybersecurity applications, particularly in resource-constrained environments such as IoT devices and autonomous robots.

## Acknowledgement

This work is funded by Program Dana Padanan 2024

## Conflict of Interest

Authors declare that there is no conflict of interests regarding the publication of the paper.

## Author Contribution

*The authors confirm their contribution to the paper as follows:* ***study conception and research planning:*** *First Author;* ***code planning and code development:*** *Second Author;* ***data validation:*** *Third Author;* ***Hardware Implementation and hardware programming:*** *Fourth Author;* ***Designing Board and System Evaluation:*** *Fifth Author;* ***Managerial and manuscript preparation:*** *Sixth and Eight Author;* ***Drafting the Manuscript and Technical Development:*** *Seven Author. All authors reviewed the results and approved the final version of the manuscript.*

## References

[1] L.S. Dalenogare, G.B. Benitez, N.F. Ayala, & A.G. Frank (2018) The expected contribution of Industry 4.0 technologies for industrial performance, *International Journal of Production Economics*, 204, 383–394, https://doi.org/10.1016/j.ijpe.2018.08.019

[2] A.I. Regla, & E.D. Festijo (2022) Performance analysis of light-weight cryptographic algorithms for Internet of Things (IoT) applications: a systematic review, *2022 IEEE 7th International Conference for Convergence in Technology (I2CT)*, Mumbai, India, 1–5, https://doi.org/10.1109/I2CT54291.2022.9824108

[3] E. Dinlersoz, & Z. Wolf (2023) Automation, labor share, and productivity: plant-level evidence from U.S. manufacturing, *Economics of Innovation and New Technology*, 33, 604–626, https://doi.org/10.1080/10438599.2023.2233081

[4] B. Yuan, M. Yang, Z. Xu, Q. Chen, Z. Song, Z. Li, D. Zou, & H. Jin (2024) Leakage of authorization-data in IoT device sharing: new attacks and countermeasure, *IEEE Transactions on Dependable and Secure Computing*, 21(4), 3196–3210, https://doi.org/10.1109/TDSC.2023.3323713

[5] Schuh, G., Anderl, R., Dumitrescu, R., Krüger, A., Hompel, M.T., & Studie, A. (2020) Industrie 4.0 maturity index. Managing the digital transformation of companies – UPDATE 2020 – (acatech STUDY). https://www.acatech.de/publikation/industrie-4-0-maturity-index-update-2020/

[6] J.T. Amael, J.E. Istiyanto, Z. Hakim, R.H. Sari, A.Z.K. Frisky, & O. Natan (2024) Securing ventilators: integrating hardware security modules with SoftHSM and cryptographic algorithms, *2024 IEEE 33rd International Symposium on Industrial Electronics (ISIE)*, Ulsan, Korea, Republic of, 1–6, https://doi.org/10.1109/ISIE54533.2024.10595781

[7] F. Zhang, H.A.E. Kodituwakku, J.W. Hines, & J.B. Coble (2019) Multilayer data-driven cyber-attack detection system for industrial control systems based on network, system, and process data, *IEEE Transactions on Industrial Informatics*, 15(7), 4362–4369, https://doi.org/10.1109/TII.2019.2891261

[8] Z. Wu, R. Liu, & H. Cao (2019) ECDSA-based message authentication scheme for BeiDou-II navigation satellite system, *IEEE Transactions on Aerospace and Electronic Systems*, 55(4), 1666–1682, https://doi.org/10.1109/TAES.2018.2874151

[9] Sousa, B., Cruz, T.R., Arieiro, M., & Pereira, V. (2021) An ELEGANT dataset with denial of service and man in the middle attacks, ArXiv, abs/2103.09380. https://doi.org/10.48550/arXiv.2103.09380

[10] G. Ramyasri, G.R. Murthy, S. Itapu, & S. Mohan Krishna (2023) Data transmission using secure hybrid techniques for smart energy metering devices, *e-Prime – Advances in Electrical Engineering, Electronics and Energy*, 4, 100134, https://doi.org/10.1016/j.eprime.2023.100134

[11] A. Abidi, B. Bouallegue, & F. Kahri (2014) Implementation of elliptic curve digital signature algorithm (ECDSA), *2014 Global Summit on Computer & Information Technology (GSCIT)*, Sousse, Tunisia, 1–6, https://doi.org/10.1109/GSCIT.2014.6970118

[12] N. Chaurasia, & P. Kumar (2023) A comprehensive study on issues and challenges related to privacy and security in IoT, *e-Prime – Advances in Electrical Engineering, Electronics and Energy*, 4, 100158, https://doi.org/10.1016/j.eprime.2023.100158

[13] S.R. Siraparapu, & S.M.A.K. Azad (2024) Securing the IoT landscape: a comprehensive review of secure systems in the digital era, *e-Prime – Advances in Electrical Engineering, Electronics and Energy*, 10, 100798, https://doi.org/10.1016/j.eprime.2024.100798

[14] I. Peradeniya, J. E. Istiyanto, & O. Natan (2025) AFPGA-Based Hardware Signature Authenticator with IC-Security for Securing Robot-to-Robot Communications, *International Conference on Control and Robotics Engineering (ICCRE)*, Nagoya, Japan, May 2025, pp. 121-125, https://doi.org/10.1109/ICCRE65455.2025.11093502

[15] S. Lamba, & M. Sharma (2013) An efficient elliptic curve digital signature algorithm (ECDSA), *2013 International Conference on Machine Intelligence and Research Advancement (ICMIRA)*, Katra, India, 179–183, https://doi.org/10.1109/ICMIRA.2013.41

[16] D. Manel, O. Raouf, H. Ramzi, & A. Mtibaa (2013) Hash function and digital signature based on elliptic curve, *14th International Conference on Sciences and Techniques of Automatic Control & Computer Engineering (STA)*, Sousse, Tunisia, 388–392, https://doi.org/10.1109/STA.2013.6783160

[17] M. Prabu, & R. Shanmugalakshmi (2009) A comparative analysis of signature schemes in a new approach of variant on ECDSA, *2009 International Conference on Information and Multimedia Technology (ICIMT)*, Jeju, Korea (South), 491–494, https://doi.org/10.1109/ICIMT.2009.65

[18] C. Singh, & A.K. Jain (2024) A comprehensive survey on DDoS attacks detection & mitigation in SDN-IoT network, *e-Prime – Advances in Electrical Engineering, Electronics and Energy*, 8, 100543, https://doi.org/10.1016/j.eprime.2024.100543

[19] S.M. Farooq, S.M. Suhail Hussain, & T.S. Ustun (2019) Elliptic curve digital signature algorithm (ECDSA) certificate based authentication scheme for advanced metering infrastructure, *2019 Innovations in Power and Advanced Computing Technologies (i-PACT)*, Vellore, India, 1–6, https://doi.org/10.1109/i-PACT44901.2019.8959967

[20] K. Ravi, & S.A. Kulkarni (2013) A secure message authentication scheme for VANET using ECDSA, *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, Tiruchengode, India, 1–6, https://doi.org/10.1109/ICCCNT.2013.6726769

[21] J. Doerner, Y. Kondi, E. Lee & A. Shelat (2018) Secure Two-party Threshold ECDSA from ECDSA Assumptions, 2018 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, pp. 980-997

[22] M.S.U.I. Sami, T. Zhang, A.M. Shuvo, & M.S.U. Haque (2024) Advancing trustworthiness in system-in-package: a novel root-of-trust hardware security module for heterogeneous integration, IEEE Access, PP(99), 1–1, https://doi.org/10.1109/ACCESS.2024.3375874

[23] V. Deshpande, L. George, & H. Badis (2019) SaFe: a blockchain and secure element based framework for safeguarding smart vehicles, 2019 12th IFIP Wireless and Mobile Networking Conference (WMNC), Paris, France, 181–188, https://doi.org/10.23919/WMNC.2019.8881408

[24] O.O. Egunjobi, A. Gomes, C.N. Egwim, & H. Morais (2024) A systematic review of blockchain for energy applications, *e-Prime – Advances in Electrical Engineering, Electronics and Energy*, 9, 100751, https://doi.org/10.1016/j.eprime.2024.100751

[25] S.S. Supase, & J.R. Pansare (2024) A secured and fault-tolerant algorithm for electing an efficient coordinator in distributed networks, e-Prime – *Advances in Electrical Engineering, Electronics and Energy*, 10, 100791, https://doi.org/10.1016/j.eprime.2024.100791

[26] J. Divya, & S. Shivagami (2020) A study of secure cryptographic based hardware security module in a cloud environment, *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, India, 1273–1279, https://doi.org/10.1109/I-SMAC49090.2020.9243328

[27] S. Yoon, B. Kim, & Y. Kang (2023) Security enhancement scheme for mobile device using H/W cryptographic module, *2023 14th International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju Island, Korea, Republic of, 1450–1452, https://doi.org/10.1109/ICTC58733.2023.10393219

[28] Nyangaresi, V.O., Al-Joboury, I.M., Al-sharhanee, K.A., Najim, A.H., Abbas, A.H., & Hariz, H.M. (2024) A biometric and physically unclonable function-based authentication protocol for payload exchanges in Internet of Drones, e-Prime – *Advances in Electrical Engineering, Electronics and Energy*, https://doi.org/10.1016/j.prime.2024.100471

[29] L.C. Evangelista Bem, B.D. Barros Brito, P.H. Pereira de Oliveira, A.B. Moura Santos, and J.C. da Silva (2023) Development of an application for the verification of electricity rates, *e-Prime – Advances in Electrical Engineering, Electronics and Energy*, 3, http://dx.doi.org/10.1016/j.prime.2023.100122

[30] A.E. Putra, O. Natan, and J.E. Istiyanto (2025) Optimizing FPGA resource allocation for SHA-3 using DSP48 and pipelining techniques, *IIUM Engineering Journal*, 26(1), 240–253, https://doi.org/10.31436/iiumej.v26i1.3328

[31] Z. Hakim, O. Natan, R. H. Sari, J. T. Amael, A. Dharmawan and J. E. Istiyanto, "6-DOF Arm Robot Control Using Open-Source FPGA," 2024 9th International Conference on Control and Robotics Engineering (ICCRE), Osaka, Japan, 2024, pp. 157-161, doi: 10.1109/ICCRE61448.2024.10589887.

[32] A.E. Putra, O. Natan, and J.E. Istiyanto (2024) Enhancing CNN deployment efficiency on mobile devices with FPGA-based accelerators: Memory-based convolution and resource optimization, *ICIC Express Letters Part B: Applications*, 15(10), 989–997, http://dx.doi.org/10.24507/icicelb.15.10.989

[33] M. M. M. Nadzri & A. Ahmad (2022) SoC FPGA-Based Rapid Prototyping of Compressed, Secured and Wireless Image Transmission for Wildlife Surveillance System, *Int. Journal of Integrated Engineering*, vol. 14, no. 4, pp. 276–285, https://doi.org/10.30880/ijie.2022.14.04.021

[34] C. W. Peng, A. B. Jambek, S. B. Dass, L. T. Wah, L. L. Laudis, A. Yadlapati, G. C. Udari, & B. R. S. Reddy (2024) Performance Evaluation of RISC-V Microcontroller System on FPGA: A Study of the NEORV32 Core, *Int. Journal of Integrated Engineering*, vol. 16, no. 1, pp. 312–317, https://doi.org/10.30880/ijie.2024.16.01.026

[35] A. Nasir, R. A. Arshah, M. R. A. Hamid, & S. Fahmy (2022) Information Security Culture Concept towards Information Security Compliance: A Comparison between IT and Non-IT Professionals, *Int. Journal of Integrated Engineering,* vol. 14, no. 3, pp. 157–165, https://doi.org/10.30880/ijie.2022.14.03.017