applied sciences

MDPI

Article

A Novel Digital Signature Scheme for Advanced Asymmetric Encryption Techniques

Farid Lalem [1], Abdelkader Laouid [2,*], Mostefa Kara [2], Mohammed Al-Khalidi [3] and Amna Eleyan [3]

[1] Research Laboratory of Applied Mathematics and Information Systems Security, Cherchell 42002, Algeria; lalemfarid@gmail.com
[2] LIAP Laboratory, Computer Science Department, University of El Oued, El Oued 39000, Algeria
[3] Department of Computing and Mathematics, Manchester Metropolitan University, Manchester M1 5GD, UK
* Correspondence: abdelkader-laouid@univ-eloued.dz

Abstract: Digital signature schemes are practical mechanisms for achieving message integrity, authenticity, and non-repudiation. Several asymmetric encryption techniques have been proposed in the literature, each with its proper limitations. RSA and El Gamal prove their robustness, but are unsuitable in several domains due to their computational complexity. Other asymmetric encryption schemes have been proposed to provide a cloud homomorphic encryption service, where the researchers focused only on how to ensure the homomorphic property. This paper proposes a new digital signature scheme dedicated to a family of encryption techniques. The proposal consists of two parts: the first focused on the secret key, and the second focused on the public key. Signature validity checking was performed by multiplying these two parts to reform again the sender's public key, then comparing the result with the decrypted message. The validation of the decrypted message guarantees data integrity, where the signer public key is used to ensure authenticity. The proposed scheme takes a shorter execution time for the entire signature operation, including signing and verification, compared to other modern techniques. The analysis showed its robustness against private key recovery and forgery attacks. The implementation results of the proposed scheme showed promising performance in terms of complexity and robustness. The results confirmed that the proposed scheme is efficient and effective for signature generation and verification.

Keywords: secure IoT communication; privacy-preserving; data integrity verification; asymmetric authentication; modern data encryption

Citation: Lalem, F.; Laouid, A.; Kara, M.; Al-Khalidi M.; Eleyan A A Novel Digital Signature Scheme for Advanced Asymmetric Encryption Techniques. *Appl. Sci.* **2023**, *13*, 5172. https://doi.org/10.3390/app13085172

Academic Editor: Luis Javier García Villalba

Received: 22 March 2023
Revised: 17 April 2023
Accepted: 18 April 2023
Published: 21 April 2023

1. Introduction

Information security has become crucial today, especially with the increasing Internet of Things (IoT) adoption. The IoT refers to a network of connected devices, sensors, and machines that can communicate with each other and exchange data over the Internet. The proliferation of IoT devices has led to an exponential increase in the number of attack vectors that cybercriminals can exploit. Many IoT devices have limited processing power and memory, making them more susceptible to security breaches. Furthermore, the vast amounts of data generated by these devices provide valuable insights into people's daily lives, making them attractive targets for hackers seeking to steal personal information.

Securing IoT devices is critical to prevent cyber attacks and protect users' privacy. The security of these devices must be designed into them from the start rather than being added as an afterthought. The use of encryption, authentication, signature, and access control mechanisms can help secure IoT devices and protect them from unauthorized access.

On the other hand, Software-Defined Networking (SDN) can provide additional security measures for IoT devices [1]. SDN can provide a centralized control plane for network traffic, monitoring, and analyzing traffic patterns and detecting anomalies that may indicate a security breach. A novel intrusion detection system for IoT was presented

in [1], based on software-defined networking and deep learning techniques. The proposed system utilizes a deep learning classifier to detect anomalies in the IoT without imposing security profiles on the IoT devices themselves. The system was implemented and tested in a simulated environment, and the results were evaluated using various performance metrics and compared against other relevant methods.

Cryptography aims to protect private information from unauthorized access, ensuring data integrity and authentication while offering other services [2–4]. The encryption method plays a crucial role in guaranteeing security by encrypting messages so that they are clear only to the sender and intended recipient [5–8]. Unfortunately, some standard encryption algorithms are difficult to implement in resource-constrained environments due to their implementation scale, heavy complexity, and power consumption [9].

To this end, lightweight encryption techniques are needed to balance the implementation cost, speed, security performance, and power consumption on resource-constrained devices, i.e., the IoT and wireless sensor networks. Lightweight cryptography is designed to be simpler and faster than traditional cryptography for securing resource-limited devices such as smart cards, radio frequency identification (RFID) tags, sensor networks, and embedded systems [9–12].

Meanwhile, lightweight asymmetric encryption techniques may not offer the same security level and functionality as more complex algorithms. However, they can still be helpful in specific contexts where performance and efficiency are critical. Hence, when selecting an asymmetric encryption technique, it is essential to carefully evaluate the trade-offs between security, functionality, and computational resources.

The digital signature is an essential element in asymmetric encryption. It offers several services, including authenticity, integrity, and non-repudiation of a digital message. Using a digital signature, the message receiver can verify its identity, guarantee that the message has not been tampered with during transmission, and prevent the denial of having sent the message. Several digital signature schemes have been proposed to offer practical mechanisms for achieving message integrity [13,14], authenticity, and non-repudiation. Digital signature schemes are used in electronic funds transfer, data exchange, and software distribution. Compared with physical signatures, they cannot be changed nor copied by someone else, and the signers cannot repudiate signatures later [15,16].

In asymmetric encryption, digital signature schemes are based on the public key. The output of the signature process is called the digital signature, a cryptographic value calculated from the data and a private key. The signer's private key is used to sign the data, while the signatory's public key is used to verify the signature by the recipient [17,18].

In cryptography, asymmetric techniques have gained significant importance due to their capability to secure digital communication channels. These techniques use different keys for encryption and decryption, and their security relies on the computational complexity of certain mathematical problems, such as factoring large numbers or discrete logarithms. However, some lightweight asymmetric techniques do not offer additional security such as the digital signature.

Despite lacking a digital signature, these lightweight asymmetric techniques are ideal for use in resource-constrained environments, such as embedded systems and mobile devices. They provide basic encryption and decryption functionality while minimizing computational resources, enabling them to be used in low-power devices with limited processing capabilities.

One common approach to creating a public key in such lightweight asymmetric techniques is to use a key of the form $k + r \times p$, where p is a large prime number and k and r are random integers. This approach has led to the development of numerous asymmetric encryption techniques, but none include a corresponding digital signature.

To address this gap, the paper proposes a technique for creating digital signatures for these lightweight asymmetric encryption techniques. The proposed method uses the private keys k, r, and p to generate a signature that can be verified using the public key $k + r \times p$. By providing a digital signature for these encryption techniques, this new method

enhances the security level and opens up new possibilities for their use in applications that require stronger security guarantees.

The rest of the paper is organized as follows: Section 2 discusses and reviews the relevant works and presents the encryption techniques that use $k + r \times p$ as a public key. Section 3 explains the proposed scheme. In Section 4, we perform an analyses against attacks. Section 5 presents the experimental results and discussions. Finally, the conclusion is presented in Section 6.

2. Related Work

The RSA signature scheme was introduced in the late 1970s and is still widely used today. It involves hashing the message to be signed and then encrypting the hash value using the sender's private key. In the 1990s, the Digital Signature Algorithm (DSA) was introduced based on the discrete logarithm problem [19]. The DSA quickly became popular and is widely used in various applications today. The early 2000s saw the introduction of the Elliptic Curve Digital Signature Algorithm (ECDSA), which is based on elliptic curve cryptography [20,21]. ECDSA is known for its small key size and fast computation time and is commonly used in resource-constrained environments. In recent years, there has been a trend towards developing lightweight digital signature schemes designed to be fast and efficient, particularly for use in resource-constrained environments such as the IoT and other embedded systems [22–24]. This section sheds some light on the recent related work in the literature.

In [25], El Gamal proposed the El Gamal signature scheme, which is one of the first digital signature schemes based on the algebraic nature of modular exponentiation. The El Gamal signature scheme involves generating a digital signature for a message using a private key and verifying the signature using a public key. The El Gamal signature scheme ensures the authenticity of data sent over insecure channels. However, the El Gamal signature algorithm has difficulty computing discrete logarithms; furthermore, it is a non-deterministic algorithm, meaning that there are many valid signatures for a given message, and the algorithm considers each of these valid signatures to be authentic [26].

In [27], Mohammed et al. added some improvements to the ElGamal signature scheme. A new blind signature scheme was proposed in their work based on the number theory operations and modular arithmetic techniques. This scheme has the advantage of ensuring greater anonymity for participants, where if a message is signed multiple times, the corresponding signatures will differ.

A new and fast cryptographic digital signature scheme was proposed in [28] by Alia et al. The vital link between Mandelbrot and Julia fractal sets was used to generate the private and public keys using their particular functions (i.e., Mandelfn and Juliafn functions). The first fractal function inputs the selected private key and generates the corresponding public key. The message is then signed with the recipient's public key using the second fractal function, and the received message is verified against the recipient's private key. In [23], Mughal et al. proposed a lightweight Shortened Complex Digital Signature Algorithm (SCDSA) based on short complex numbers for both signature and verification operations. This algorithm aims to secure the communication between user-centric IoT devices and mitigate the communication and computational overhead on the network. Due to the multi-option parameter selection mechanism, their scheme is secure enough to resist traffic analysis attacks and has better resilience than extensive operations in DSA-based schemes.

Johnson et al. presented an Elliptic Curve Digital Signature Algorithm (ECDSA) in [21] based on the curve drawn by a mathematical equation. On the curve, a point is selected randomly and considered its point of origin. Then, a random number is generated. It is precisely this random number that is known as the private key. Then, combining the private key and the origin point using a specific equation, a second point on the curve is obtained and considered the public key. This process is considered safe, and it is only possible to establish the mathematical relationship from the private key to the public key, but not the opposite. The ECDSA is used across many security systems and is the basis of Bitcoin

security. The ECDSA provides higher security with shorter key lengths than many popular algorithms such as the RSA and DSA. However, this algorithm is unsuitable for constrained devices that are limited in resource computation. To improve the ECDCA's computational cost, Junru in [29] proposed two schemes named ECDSA 1 and ECDSA 2. Junru's goal is to improve the ECDCA's computational cost while keeping the same security level as compared to the ECDSA. ECDSA 1 is suitable for constrained devices on the signatory's side, whereas ECDSA 2 is relevant for constrained devices on the verifier's side.

In [30], Lavanya and Natarajan designed a Lightweight Digital Signature Algorithm (LWDSA) based on the ECDSA and cryptographic hash functions dedicated to WSNs. In this modified the ECDSA, the major variation is using the MBLAKE2b hash function instead of SHA-1 to generate the hash code. The LWDSA outputs have been compared with the traditional authentication process of the ECDSA. The analysis of the QoS parameters such as the latency, energy, throughput, etc., were performed over a 30-node network simulated in NS-2. Their simulation performed better by reducing the computation time and energy consumption, increasing the network lifetime. However, the computation time has not been clearly indicated for the generation and verification of the signature process. In addition, it is challenging to measure QoS parameters for a large-scale and dynamic WSN network in order to validate the proposed scheme.

Kavin et al. aimed to develop an Enhanced Digital Signature Algorithm (EDSA) in [31] to ensure the integrity and secure access of the data stored in the cloud. The idea of EDSA focuses on elliptic curve square points generated by improved equations. The proposed EDSA generates two elliptic curves by applying the improved equations, which are used as public keys to perform the signing and verification process. In addition, a new basic formula is introduced for performing digital signature operations such as signing, verifying, and comparing. A compression technique is used to reinforce reducing the bit size of the signature.

A new digital signature scheme based on Iterated Function Systems (IFSs) was designed in [32] by Al-Saidi et al. Fractals were used to design the digital signature system based on IFS transformations. The receiver requests a signature from the signer, who then delivers a fractal signature to the requester without knowing the content of the message. The proposed scheme includes initialization, signing, extraction, and verification. The advantage of the fractal is that only a few parameters must be stored. The fractal scheme outperforms the DSA and RSA schemes due to the lower computational overhead and smaller key sizes. However, the scheme still incurs high computational overhead for constrained devices and needs to be adapted to such device types.

In [33], Wei et al. proposed a new attribute-based proxy signature scheme on a lattice that can resist quantum attacks. The scheme has the properties of both attribute-based signatures and proxy signatures providing fine-grained access control. The scheme allows the original signer to sign a warrant using their attribute keys to delegate signing authority. Then, the proxy signer signs the message only when the attribute set of the original signer satisfies the access structure and the warrant is valid. The verifier also checks the original signer's attributes and the validity of the signature. This feature provides fine-grained access control; however, the main disadvantage of lattice-based constructions is that they generally involve operations on, and storage of, large $n \times n$ matrices. This resulted in the schemes being rather inefficient and unsuitable for practical use.

Zhou et al. proposed a mixed private and consortium-based blockchain in [34] to realize the secure storage of medical cyber physical data. In the system, a threshold signature system based on blockchain was proposed for joint consultation. Using the security and threshold properties of the threshold signature, the treatment can be carried out when the threshold number is reached, and medical data can be uploaded to the consortium blockchain. Despite being application-domain-dependent, the security analysis and performance analysis showed that the scheme has advantages in safety and performance and is suitable for the medical environment to a certain extent.

Within the medical data domain as well, Al-Zubaidie et al. developed the Pseudonymization and Anonymization with the XACML (PAX) modular system in [35]. The proposed system uses a random pseudonym to separate personal information about patients' data, anonymity to hide subjects' information, and XACML to create distributed access control policies to authorize subjects' requests to objects' records in Electronic Health Records (EHRs). It provides a security and privacy solution to the problem of safe-access decisions for patients' data in the EHR. The results of theoretical and experimental security analysis proved that PAX provides security features in preserving the privacy of healthcare users and is safe against known attacks. However, the validity of the solution has not been demonstrated for other application domains.

Encryption Techniques that Use $k + r \times p$ as a Public Key

In asymmetric encryption, there are two keys: a public key (PcK) for encryption and another private key (PvK), which is used for decryption. Any user can send a message encrypted by PcK to its owner, and only the owner of PvK can decrypt this message. In addition to encryption, PvK could play another vital role: message signing. In general, when sending a message, a user must sign it using his/her PvK for two reasons: to ensure message integrity and authenticity.

Many schemes use the RSA as a signature system, and this forces these techniques to use two private keys and two public keys, i.e., two pairs (PvK, PcK) and (e, d), where (PvK, PcK) are the keys used in the encryption/decryption processes and (e, d) are the RSA keys used for signing. Otherwise, these techniques must use encryption/decryption keys that respect the RSA conditions $\phi(n) = (p-1) \times (q-1)$ and $d \times e = 1 \mod \phi(n)$. In the RSA and elliptic curve, it is difficult or impossible because their keys may have a specific form and/or satisfy particular conditions. Therefore, some solutions propose using a signature scheme compatible with their key pair (PvK, PcK).

Among the asymmetric encryption techniques that do not have—to the best of our knowledge—a signature scheme compatible with their key pair are techniques that use a public key of the following form:

$$pk = k + r \times p \tag{1}$$

where pk is the public key, k and p are the private key, and r is an integer used to mask private keys and generate encryption noises. In addition, the public primitive $n = p \times q$ with the trapdoor p and q, which are two large prime numbers.

It is essential here to mention that r is not necessarily private, and it can be published where the adversary knows (pk, n):

$$n = p \times q \tag{2}$$

We note here that there are two equations ((1) and (2)) and three unknown numbers (k, p, and q); then, it is challenging with large p and q to obtain the private values k and p as long as there is no mathematical relation between k and p ($p \neq f(k)$). r verifies $Enc_{pk}(m) > n$, whose goal is to generate a noise greater than n. Table 1 illustrates some encryption schemes that use $k + r \times p$ as a public key.

In [36,37], the authors used a symmetric scheme; however, we mean here the asymmetric version of this technique where $c = m + pk \mod n$ and $c = (m + r \times pk) \mod k \times p$. Defining a signature scheme for these kinds of lightweight asymmetric techniques is challenging. In the next section, we will explain, in detail, the proposed digital signature scheme dedicated to the techniques that use $k + r \times p$ as a public key.

Table 1. Schemes that use $k + r \times p$ as a public key.

Scheme	Enc	Dec
[38]	$c = m \times pk$	$m = (c - c \mod \alpha) \mod (\alpha - 1)$
[39]	$c = d_0 \times pk_0 + d_1 \times pk_1 + ..d_i \times pk_i$	$m = \sum_{j=i}^{0} \frac{c}{k_j} \times 10^j$, $c \leftarrow c - \frac{c}{k_j} \times k^j$
[40]	$c = (m + r \times (pk - 1)) \mod n$	$m = (c \mod p) \mod (k - 1)$
[36]	$c = (m + k) \mod p$	$m = (c - k) \mod p$
[37]	$c = (m + r \times k) \mod k \times p$	$m = c \mod p$

3. The Proposed Signature Scheme

In asymmetric encryption techniques, the receiver needs to authenticate the author of sent data and guarantee its non-repudiation, which is ensured by digital signature.

In fact, there are many algorithms that are used to create signatures, and these algorithms use smaller keys, for example Elliptic-Curve-Cryptography (ECC)-based signature algorithms, but to use ECC, both the private and public keys must be created over an elliptic curve. In this paper, we studied the creation of an accompanying signature for cryptographic algorithms that have a linear and very special public key generated independently of ECC, where $pk = k + r \times p$, with p a prime number and k and r random numbers.

To the best of our knowledge, there is still no corresponding signature for this form of keys. We focused on this type of public key for its ease of creation and also for its resistance to attacks so the secret parameters (k, r, and p) cannot be extracted from it.

To provide a suitable scenario describing the new signature scheme's functional mode, we need certain details about the planned architectural model to use. Figure 1 shows the proposed architectural model. Most modern cryptographic architectures follow a layered approach, with each layer providing a specific security function. The proposed architecture layers include the following components: the sender, the encryption operation, the signing module, the Internet, the receiver, the decryption operation, the verifying module, and the comparing module:

1. Sender: This plays the role of the signer and transforms the plain text into a cipher text that will be used later for data validity.
2. Key generation: This component is responsible for providing all keys required for the signing and verifying processes, including private keys, split public keys, and whole public keys.
3. Signing module: By applying the newly proposed signing formulas, this component creates a digital signature of the original message that is sent by the signer. It uses private keys for extracting the two parts of the signature. The created digital signature will be sent along with the encrypted message to the Internet for further processing
4. Internet: Its role is to transport information to the receiver directly or to the cloud for storing, allowing the authorized user to access the stored data.
5. Receiver: This obtains the authorization for accessing the specific data, which are transported by the Internet and stored in an encrypted format with a signature. The encrypted message will be accessed by the receiver and decrypted using the decryption operation. The receiver then has the original data and a digital signature created by the sender using the sender's private key. The authorized receiver can then verify the originality of the message with the help of the sender's public key.
6. Decryption operation: This transforms the cipher text into plain text.
7. Verifying module: The verification module verifies the sent digital signature of the original message that is created by the sender. New values will be created by applying the verification formula.
8. Comparing module: The created values using the signature, which is available along with the original message, will be compared to each other. If both values are the same, then the integrity of the data is validated; otherwise, the accessed data are a modified message or a fake message.

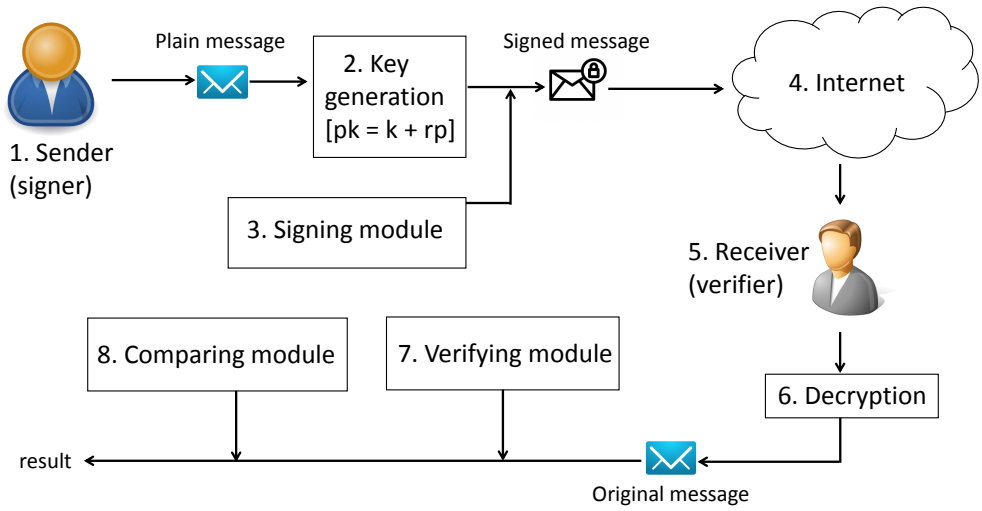

Figure 1. The proposed architectural model.

3.1. The Proposed Scheme

The proposed DS technique uses the private key k and the secret trapdoor p of the sender to generate a DS represented in two parts s_1 and s_2, which are used for performing the signing and verifying processes. Moreover, new base formulas are also introduced for conducting digital signature operations such as signing and verifying. Our proposed protocol is explained under three subsections including the signing, verification, and comparison processes. However, first, let us define the following primitives:

- The sender integer random r split into two other random numbers r_1 and r_2 where $r = r_1 \times r_2$.
- The sender private key is $K_s = (k_s, r_2)$,
- The sender public key is $PK_s = (pk_s, n, n', r_1)$, where $pk_s = k_s + r \times p$, $n = p \times q$ with p and q two large prime numbers; n' is a small modulus (160 bits).
  To generate a secure public key, the integer r must verify $r > q$. Furthermore, p and q must be secure primes, i.e., they are of the form $2 \times p' + 1$ and $2 \times q' + 1$, respectively, with p' and q' also primes:
- The plain message m is encrypted by the encryption function Enc using receiver public key pk_r, $Enc_{pkr}(m) = c$.
- The plain message m is signed by the signing function Sig() using sender private keys (k_s, r_2, p).
- The receiver private key is k_r
- The cipher message c is decrypted by the decryption function Dec using receiver private key k_r, $Dec_{kr}(c) = m$.
- The received signature Sig is verified by the verifying function Verf() using sender public keys (pk_s, r_1).

To imply the public key ($pk = k + r \times p$) in the signature, the main idea of our proposal consists of two steps: First, split the signature into two parts s_1 and s_2; one uses k and the other uses $r \times p$. Second, split randomly r into r_1 and r_2, where $r = r_1 \times r_2$; keep one secret (r_2), and the other is public. By this, we will obtain $s_1 = m^k$ and $s_2 = m^{r_2 \times p}$. The receiver can easily first compute $x = s_1 \times s_2^{r_1}$, which is equal to $m^{k+r \times p}$, and secondly compute $y = m^{pk}$; finally, they are compared. The proposed protocol is explained below:

3.1.1. Signing Process

In the signing process, we need to go through the encryption process, as the cipher text will be used in the verification process to extract the original message and ensure the data integrity. To sign a message m, the sender encrypts m as:

$$c = Enc_{pkr}(m) \tag{3}$$

Then, the sender signs the message m by calculating s_1 and s_2, which contain their private keys k_s, r_2, and p as follows:

$$s_1 = m^{k_s} \mod n' \tag{4}$$

and

$$s_2 = m^{r_2 \times p} \mod n' \tag{5}$$

The digital signature to be sent to the receiver is sig_m:

$$sig_m = (s_1, s_2) \tag{6}$$

The workflow of a signing process in the proposed DS technique is shown in Algorithm 1.

Algorithm 1 Signing algorithm.

Require: message m, secret keys (k, r_2, p), public keys (n, n', pk_r)
Ensure: sig_m

1: **function** SIG
2: compute $s_1 : s_1 \leftarrow m^{k_s} \mod n'$
3: compute $s_2 : s_2 \leftarrow m^{r_2 \times p} \mod n'$
4: $sig_m \leftarrow (s_1, s_2)$
5: **return** sig_m
6: **end function**

3.1.2. Verifying Process

After decrypting the cipher text c and retrieving the original message $m = Dec_{kr}(c)$, the receiver should calculate the value of x using the sender's public key pk_s as follows:

$$x = m^{pk_s} \mod n' \tag{7}$$

where $pk_s = k_s + r \times p$

The receiver uses the computed value of m to detect any manipulation of the cipher text and ensure data integrity. The use of the sender's public key ensures authenticity. Then, the check value y will be calculated:

$$y = (s_1 \times s_2^{r_1}) \mod n' \tag{8}$$

Then comes the comparison process:

$$x \stackrel{?}{=} y \tag{9}$$

The workflow of a verification process in the proposed DS scheme is shown in Algorithm 2, which ensures validity (integrity and authenticity).

3.1.3. Correctness

The sender calculates s_1 and s_2 using his/her private keys, where $s_1 = m^{k_s}$ and $s_2 = m^{r_2 \times p}$; the receiver recovers m and calculates $x = m^{pk_s}$, where $pk_s = k_s + r \times p$ and

$r = r_1 \times r_2$. Then, the receiver calculates $y = (s_1 \times s_2^{r_1}) \longrightarrow y = m^{k_s} \times (m^{r_2 \times p})^{r_1}$; this implies $y = m^{k_s} \times m^{r_1 \times r_2 \times p} = m^{k_s} \times m^{r \times p}$; $\longrightarrow y = m^{k_s + r \times p} = m^{pk_s}$; this gives $y = x$.

Algorithm 2 Verification algorithm.

Require: cipher text c, signature sig_m, public keys (pk_s, n, n', r_1), private key k_r
Ensure: decryption and message integrity

```
function VERF
    compute m : m ← Dec_{k_r}(c)
    compute x : x ← m^{pk_s} mod n'
    compute y : y ← (s_1 × s_2^{r_1}) mod n'
    if x = y then
        return m
    else
        return invalid
    end if
end function
```

Therefore, the proposal is correct and the data receiver can easily verify sender authenticity using the sender's public key.

3.1.4. A Practical Example

To show how the proposed model works, we present the following example: prime number p of size equal to 1 kbits and prime number q of size equal to 1 kbits; this gave us a modulus $n = p \times q$ of size equal to 2 kbits. We chose a secret key k of size equal to 0.5 kbits and a random number r of size equal to 1 kbits. This gave us a public key pk of size equal to 2 kbits. Both sub-keys r_1 and r_2 are of size equal to 0.5 kbits. The message m is equal to 7 bits. After calling the Sig function, we obtain both s_1 and s_2 of size equal to 160 bits. After calling the $Verf$ function, we obtain the values of x and y (both of size equal to 160 bits). Finally, we found $x = y$, demonstrating the proposed scheme's validity and applicability.

We note that the value of x is equal to the value of y, knowing that y is calculated by using the split sender's public key r_1 and computed m, and x is calculated by using the whole sender's public key pk_s.

4. Analysis

The most-important security aspect of the signature scheme is retrieving the private key or forging the signature process, which we will discuss in the following.

4.1. Attacks Aiming to Recover the Private Key from the Public Key

In our proposal, the modulus n is public. The proposed scheme is based on a Factorization Problem (FB), where the adversary has to factorize $n = p \times q$ to obtain p.

If there is a successful attack, the adversary calculates the secret key k from pk, where

$$k = pk \mod p \tag{10}$$

After calculating k, the adversary can extract the secret number r_2 from r since r_1 is public, where

$$r = (pk - k) \div p \tag{11}$$

Then, $r_2 = r \div r_1$. Now, the adversary has the whole private key (k, r_2, p) and can easily produce a valid signature.

Therefore, secure prime numbers must be used; a prime number p is called safe if $p = 2 \times p' + 1$, such that p' is also a prime number. In FB, a size of p and q that equals 1024 bits is considered secure. To achieve a valid decryption, a message m to be encrypted

must be less than the private key k, so the size of k usually depends on the used encryption technique and the plain text range.

4.2. Attacks Aiming to Recover the Private Key

In this attack, the adversary tries to obtain one of the private keys k, r_2, or p that were used to create a valid signature. Signature parts $s_1 = m^{k_s} \mod n'$ and $s_2 = m^{r_2 \times p} \mod n'$ will be sent through an insecure channel, so we suppose that an adversary can obtain both s_1 and s_2.

In a successful attack, we assumed that the adversary can have s_1 and s_2, either by the interception or by being a malicious receiver. We have $s_1 = m^k$; if the adversary is a malicious receiver, he/she can also know m by decrypting $c = Enc(m)$ using its private key. Therefore, to recover k, the adversary has to solve the Discrete Logarithm Problem (DLP).

After computing k, the adversary can extract $r_2 \times p$ using the public keys (pk, r_1), where

$$r_2 \times p = (pk - k) \div r_1 \tag{12}$$

Now, the adversary has k and $r_2 \times p$; thus, he/she can create a valid signature. The same for $s_2 = m^{r_2 \times p}$, the adversary must solve the DLP in order to recover directly $r_2 \times p$. Otherwise, the adversary must solve the factorization problem in order to obtain p and then k from the public n.

We would like to point out here that r does not have to be secret. We assumed this in the proposed scheme just to increase the level of security; this means that both r_1 and r_2 can be known for everyone. Suppose, for example, that the adversary has r_1 and r_2; he/she still has to know k and p in order to create a valid signature ($s_1 = m^{k_s}$ and $s_2 = m^{r_2 \times p}$).

Knowing r_1 and r_2 does not affect the robustness of the public key ($pk = k + r \times p$), nor does it give any information about k or p. Likewise, knowing r_2 does not affect the confidentiality of s_2, nor does it give any information on the plaintext m or the private keys (k, p). Assuming that the receiver has r_2, after decrypting c and computing m, he/she will have $m^{r_2 \times p} = m'^p$, where $m' = m^{r_2}$; he/she still has to solve the DLP to obtain p. Therefore, we have hidden r only because there is no need to make it public.

4.3. Attacks for Forging Signatures

We now considered the possibility of forging a given signature without knowing the private keys k and p. In order to forge the signature of a message m, the adversary must simulate the two signature parts s_1 and s_2; because he/she does not know either k or r_2 or p, he/she must randomly generate two numbers $s'_1 = random_1$, $s'_2 = random_2$ or use random numbers k', $r2'$, and p' as secret keys, then calculate $s'_1 = m^{k'} \mod n'$ and $s'_2 = m^{r'_2 \times p'} \mod n'$. Of course, the adversary can encrypt any message m, because he/she knows the public key of the victim, where $c = Enc_{victim_{pk}}(m)$. The digital signature to be sent to the receiver is sig'_m, where $sig'_m = (s'_1, s'_2)$.

In the verifying step, the receiver will calculate x using the decrypted message m and victim public key pk_s as follows: $x = m^{pk_s} \mod n'$. To generate $s_2^{r \times p}$, the receiver should calculate $s_2^{r_1}$.

We have now $x = m^{k_s + r \times p} = m^{k_s} \times m^{r \times p}$ and $y = s'_1 \times s_2'^{r_1}$ which implies $y = m^{k'} \times (m'^{r'_2 \times p'})^{r_1}$. Thus, $y = m^{k'} \times m^{r_1 \times r'_2 \times p'}$. We can clearly see that $m^{k'} \neq m^k$, while $k' \neq k$, $m^{r_1 \times r'_2 \times p'} \neq m^{r_1 \times r_2 \times p}$, while $r'_2 \neq r_2$ and $p' \neq p$.

The adversary can easily compute xm^{pk}, then obtain two numbers α and β where $x = \alpha \times \beta$. α simulates s_1, and β simulates $s_2^{r_1}$; knowing that r_1 is public, the adversary's issue is to find a number γ that verifies $\gamma^{r_1} \mod n' = \beta$. This is equivalent to the DLP.

Therefore, to forge the proposed signature scheme, the adversary has to resolve the DLP.

5. Experimental Results

Table 2 represents the sizes used in each test. The message size is fixed in all tests where $size(m)$ = 8 bits. In this type of public key $k + r \times p$, generally $size(p) = 2 \times size(k)$ because

$k^2 < p$. r must satisfy the condition $r > q$; to minimize the complexity, we can choose $r > q$ with $size(r) = size(q)$; for example, r = 725, q = 307. We have $size(k + r \times p) = size(r \times p)$; thus, the size of the public key is $size(pk) = size(r) + size(p)$. While $n = p \times q$ and $size(r) = size(q)$, then $size(n) = 2 \times size(p)$. We note here that, for security measures, we selected p and q with similar, but not equal sizes to prevent the square computation on n. The size of s_1 and s_2 means the average size of many tests. However, we note that the sizes of s_1 and s_2 are always almost equal to the size of n.

Table 2. Parameter sizes in the different tests.

Test/Size (bits)	k	r	p	n	s_1	s_2
T_1	128	256	256	512	512	512
T_2	256	512	512	1024	1024	1024
T_3	512	1024	1024	2048	2048	2048
T_4	1024	2048	2048	4096	4096	4096

Figure 2 shows the execution time in each test, that is the Signing Process Time (SPT) and the Verification Process Time (VPT). In T_1, the sizes used are impractical, but used to show the increase in time. In T_2, T_3, and T_4, we noticed that SPT and VPT take a constant ratio between them, where $VPT = 1.33 \times SPT$. SPT and VPT increase more rapidly in T_4. Of course, the sizes used in T_4 exceed the safety recommendations, because the recommended size of primitive p is 1024 bits, which is used in T_3.

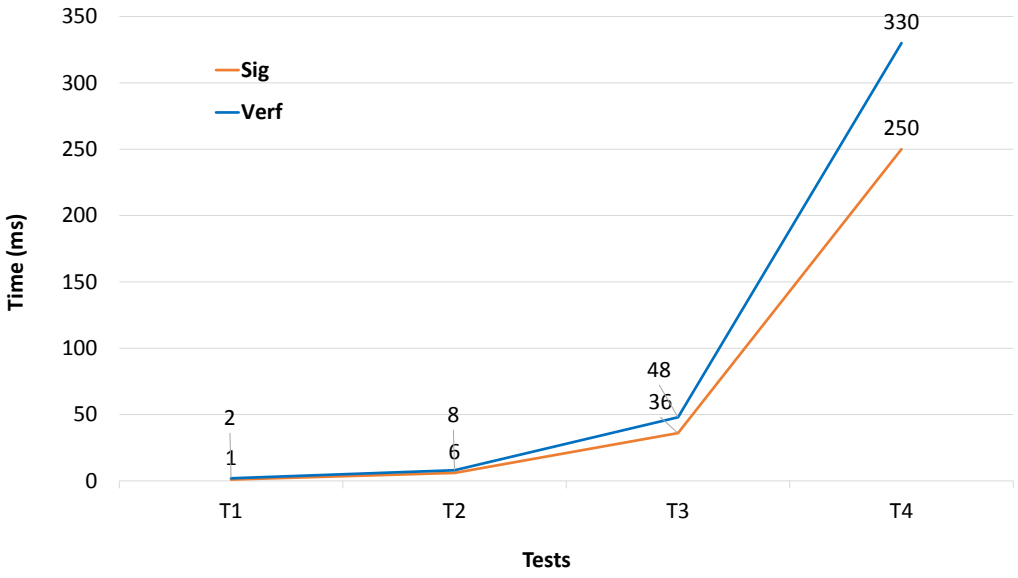

Figure 2. Signing and verifying execution time in the proposed scheme.

The results shown in this section, namely signing and verifying time, showed the applicability of the proposed method. Four experiments were presented containing realistic example values; the implementation time was not exceeded, at the highest estimate, 250 milliseconds for signing and 330 milliseconds for verifying represented in Experiment No. 4.

Table 3 shows the used notations and the corresponding runtimes for each operation.

Table 3. The used notations with runtimes.

Notation	Description	Value (ms)
T_m	multiplication	4.4
T_a	addition	2.2
T_h	hash function	19
T_e	exponentiation	11
T_r	random number generation	11
T_c	concatenation	0.9

A comparative resume of the computational costs and communication costs for the four techniques [23,26,31,41,42] and ours is illustrated in Table 4 and Figure 3.

Table 4. A comparison of the computation cost and size.

Scheme	Signing	Verification	Total (ms)	Sig (bit)	Suitable for $pk = k + r \times p$
[23]	$T_{r,c,h,m,a,e}$	$T_{2e,c,h,m}$	94.8	320	No
[31]	$T_{h,5m,2a}$	$T_{h,4m,2a}$	86.4	64–512	No
[26]	$T_{r,m,3a,e}$	$T_{2m,a,e}$	44	1024	No
[41]	$T_{r,3m,h,3c,a}$	$T_{h,3c,3m,a}$	87.4	340	No
[42]	$T_{r,3m,4c,h,a}$	$T_{a,m}$	55.6	992	No
Ours	T_{2e}	$T_{2e,m}$	48.4	320	Yes

Table 4 shows the preference of the proposed scheme compared to other schemes, where the comparison was based on the full time to perform the signature and verification operations, the signature size, and scheme suitability for systems that use a public key of the form $pk = k + r \times p$. The only scheme with less execution time is that of [26] because it uses only two exponentiation instead of four and does not use a hash function, so it achieved four milliseconds less time. The signature size in the scheme [23] was equal to our scheme size (size = $| s_1 | + | s_2 | = 2 \times | n' | = 2 \times 160 = 320$ bits). As for the rest of the schemes, the size of the signature was greater than that. All of these techniques are not suitable for cryptosystems that use a public key $pk = k + r \times p$, except for the proposed technique.

Figure 3. Comparison of computational and communication costs of our proposal with Mughal 2018 [23], Kavin 2021 [31], Zaghian 2013 [26], Ullah 2020 [41], and Ali 2020 [42] schemes.

This practical comparison, which was presented in this section, shows the effectiveness of the proposed protocol, not only in a theoretical study, but also in an applied study, outperforming many modern techniques, both in execution time and in signature size. The results of this study have a significant impact on the field of research and the field of creating digital signatures, due to the new and impressive results it presents.

6. Conclusions

In conclusion, our research presented an innovative digital signature scheme that is both robust and efficient, outperforming other modern techniques in terms of signature size and execution time. The proposed scheme provides a novel digital signature for asymmetric encryption techniques, utilizing a public key in the form of $k + r \times p$, where both k and p are private keys used in both the decryption and signature processes. Through our analysis of the proposed scheme, we demonstrated its robustness against private key recovery attacks and forgery attacks.

Furthermore, we presented a real-life signature example and conducted a series of tests with various input sizes to showcase the execution time of both the signature and verification processes. By comparing our proposed scheme with other modern techniques, we established that our approach boasts a smaller signature size of only 320 bits, thanks to the use of a small-sized public key. Additionally, we achieved a faster execution time of only 48.4 ms for the whole signature operation, including signing and verification, due to the fewer operations used by our technique.

Looking towards future directions, we aim to develop an aggregate signature protocol based on our proposal, allowing for two or more digital signatures to be joined into a single, shorter signature. This would allow different signers to generate signatures using our presented scheme.

Author Contributions: Formal analysis, F.L.; methodology, M.K.; supervision and validation, A.L.; writing—review and editing, M.A.-K. and A.E.; Data Curation and Visualisation, A.E. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No data is available.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wani, A.; Khaliq, R. SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL). *CAAI Trans. Intell. Technol.* **2021**, *6*, 281–290. [CrossRef]
2. Amalraj, A.J.; Jose, J.J.R. A survey paper on cryptography techniques. *Int. J. Comput. Sci. Mob. Comput.* **2016**, *5*, 55–59.
3. Chandra, S.; Paira, S.; Alam, S.S.; Sanyal, G. A comparative survey of symmetric and asymmetric key cryptography. In Proceedings of the 2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE), Hosur, India, 17–18 November 2014; pp. 83–93.
4. Chen, Z. Research on internet security situation awareness prediction technology based on improved RBF neural network algorithm. *J. Comput. Cogn. Eng.* **2022**, *1*, 103–108.
5. Verma, R.; Kumari, A.; Anand, A.; Yadavalli, V. Revisiting shift cipher technique for amplified data security. *J. Comput. Cogn. Eng.* **2022**. [CrossRef]
6. Kumari, S.; Yadav, R.J.; Namasudra, S.; Hsu, C.H. Intelligent deception techniques against adversarial attack on the industrial system. *Int. J. Intell. Syst.* **2021**, *36*, 2412–2437. [CrossRef]
7. Gutub, A. Boosting image watermarking authenticity spreading secrecy from counting-based secret-sharing. *CAAI Trans. Intell. Technol.* **2022**. [CrossRef]
8. Namasudra, S.; Roy, P. A new table based protocol for data accessing in cloud computing. *J. Inf. Sci. Eng.* **2017**, *33*, 585–609.
9. Buchanan, W.J.; Li, S.; Asif, R. Lightweight cryptography methods. *J. Cyber Secur. Technol.* **2017**, *1*, 187–201. [CrossRef]
10. Al-Husainy, M.A.F.; Al-Shargabi, B.; Aljawarneh, S. Lightweight cryptography system for IoT devices using DNA. *Comput. Electr. Eng.* **2021**, *95*, 107418. [CrossRef]

11. Yang, Z.; Jin, C.; Tian, Y.; Lai, J.; Zhou, J. Lis: Lightweight signature schemes for continuous message authentication in cyber-physical systems. In Proceedings of the 15th ACM Asia Conference on Computer and Communications Security, Taipei, Taiwan, 5–9 October 2020; pp. 719–731.
12. Pavithran, P.; Mathew, S.; Namasudra, S.; Singh, A. Enhancing randomness of the ciphertext generated by DNA-based cryptosystem and finite state machine. *Clust. Comput.* **2023**, *26*, 1035–1051. [CrossRef]
13. Kara, M.; Laouid, A.; Hammoudeh, M. An Efficient Multi-Signature Scheme for Blockchain. Cryptology ePrint Archive, Paper 2023/078. 2023. Available online: https://eprint.iacr.org/2023/078 (accessed on 1 February 2023).
14. Kara, M.; Laouid, A.; Dabbas, O.A.; Hammoudeh, M.; Bounceur, A. One Digit Checksum for Data Integrity Verification of Cloud-Executed Homomorphic Encryption Operations. Cryptology ePrint Archive, Paper 2023/231. 2023. Available online: https://eprint.iacr.org/2023/231 (accessed on 1 February 2023).
15. Gaborit, P.; Girault, M. Lightweight code-based identification and signature. In Proceedings of the 2007 IEEE International Symposium on Information Theory, Nice, France, 24–29 June 2007; pp. 191–195.
16. Aki, S. Digital signatures: A tutorial survey. *Computer* **1983**, *16*, 15–24. [CrossRef]
17. Katz, J. Digital signatures: Background and definitions. In *Digital Signatures*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 3–33.
18. Wong, C.K.; Lam, S.S. Digital signatures for flows and multicasts. In Proceedings of the Sixth International Conference on Network Protocols (Cat. No. 98TB100256), Austin, TX, USA, 13–16 October 1998; pp. 198–209.
19. Harn, L. New digital signature scheme based on discrete logarithm. *Electron. Lett.* **1994**, *30*, 396–398. [CrossRef]
20. Al-Zubaidie, M.; Zhang, Z.; Zhang, J. Efficient and secure ECDSA algorithm and its applications: A survey. *arXiv* **2019**, arXiv:1902.10313.
21. Johnson, D.; Menezes, A.; Vanstone, S. The elliptic curve digital signature algorithm (ECDSA). *Int. J. Inf. Secur.* **2001**, *1*, 36–63. [CrossRef]
22. Yavuz, A.A.; Ozmen, M.O. Ultra lightweight multiple-time digital signature for the internet of things devices. *IEEE Trans. Serv. Comput.* **2019**, *15*, 215–227. [CrossRef]
23. Mughal, M.A.; Luo, X.; Ullah, A.; Ullah, S.; Mahmood, Z. A lightweight digital signature based security scheme for human-centered Internet of Things. *IEEE Access* **2018**, *6*, 31630–31643. [CrossRef]
24. Suhail, S.; Hussain, R.; Khan, A.; Hong, C.S. On the role of hash-based signatures in quantum-safe internet of things: Current solutions and future directions. *IEEE Internet Things J.* **2020**, *8*, 1–17. [CrossRef]
25. Elgamal, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory* **1985**, *31*, 469–472. [CrossRef]
26. Zaghian, A.; Mansouri, M. A new blind signature scheme based on improved ElGamal signature scheme. *Int. J. Inf. Commun. Technol. Res. IJICTR* **2013**, *4*, 61–65.
27. Mohammed, E.; Emarah, A.; El-Shennawy, K. A blind signature scheme based on ElGamal signature. In Proceedings of the IEEE/AFCEA EUROCOMM 2000. Information Systems for Enhanced Public Safety and Security (Cat. No. 00EX405), Minufiya, Egypt, 24–24 February 2000; pp. 51–53.
28. Alia, M.A.; Samsudin, A.B. A new digital signature scheme based on Mandelbrot and Julia fractal sets. *Am. J. Appl. Sci.* **2007**, *4*, 850–858.
29. Junru, H. The improved elliptic curve digital signature algorithm. In Proceedings of the 2011 International Conference on Electronic & Mechanical Engineering and Information Technology, Harbin, China, 12–14 August 2011; Volume 1, pp. 257–259.
30. Lavanya, M.; Natarajan, V. LWDSA: Light-weight digital signature algorithm for wireless sensor networks. *Sādhanā* **2017**, *42*, 1629–1643. [CrossRef]
31. Kavin, B.P.; Ganapathy, S. A new digital signature algorithm for ensuring the data integrity in cloud using elliptic curves. *Int. Arab. J. Inf. Technol.* **2021**, *18*, 180–190.
32. Al-Saidi, N.; Md Said, M.R. Improved digital signature protocol using iterated function systems. *Int. J. Comput. Math.* **2011**, *88*, 3613–3625. [CrossRef]
33. Wei, L.; Li, D.; Liu, Z. Provable Secure Attribute-Based Proxy Signature Over Lattice Small Integer Solution Problem in Random Oracle Model. *Electronics* **2023**, *12*, 1619. [CrossRef]
34. Zhou, X.; Huang, J.; Chen, F.; Tang, Y.; Wang, C. A Decentralized Threshold Signature Scheme of Blockchain-Based Medical Cyber Physical Systems. Researchsquare ePrint Archive. 2021. Available online: https://assets.researchsquare.com/files/rs-869835/v1_covered.pdf? (accessed on 1 February 2023).
35. Al-Zubaidie, M.; Zhang, Z.; Zhang, J. PAX: Using pseudonymization and anonymization to protect patients' identities and data in the healthcare system. *Int. J. Environ. Res. Public Health* **2019**, *16*, 1490. [CrossRef] [PubMed]
36. Castelluccia, C.; Mykletun, E.; Tsudik, G. Efficient aggregation of encrypted data in wireless sensor networks. In Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, San Diego, CA, USA, 17–21 July 2005; pp. 109–117.
37. Yagoub, M.A.; Laouid, A.; Kazar, O.; Bounceur, A.; Euler, R.; AlShaikh, M. An adaptive and efficient fully homomorphic encryption technique. In Proceedings of the 2nd International Conference on Future Networks and Distributed Systems, Amman, Jordan, 26–27 June 2018; pp. 1–6.

38. Kara, M.; Laouid, A.; Bounceur, A.; Hammoudeh, M.; Alshaikh, M. Perfect confidentiality through unconditionally secure homomorphic encryption Using OTP With a single pre-shared key. *J. Inf. Sci. Eng.* **2023**, *39*, 183–195.
39. Chait, K.; Laouid, A.; Laouamer, L.; Kara, M. A multi-key based lightweight additive homomorphic encryption scheme. In Proceedings of the 2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP), El Oued, Algeria, 20–21 November 2021; pp. 1–6.
40. Kahla, M.E.; Beggas, M.; Laouid, A.; Kara, M.; AlShaikh, M. Asymmetric image encryption based on twin message fusion. In Proceedings of the 2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP), El Oued, Algeria, 20–21 November 2021; pp. 1–5.
41. Ullah, S.S.; Ullah, I.; Khattak, H.; Khan, M.A.; Adnan, M.; Hussain, S.; Amin, N.U.; Khattak, M.A.K. A lightweight identity-based signature scheme for mitigation of content poisoning attack in named data networking with internet of things. *IEEE Access* **2020**, *8*, 98910–98928. [CrossRef]
42. Ali, I.; Lawrence, T.; Li, F. An efficient identity-based signature scheme without bilinear pairing for vehicle-to-vehicle communication in VANETs. *J. Syst. Archit.* **2020**, *103*, 101692. [CrossRef]