Journal of *Cybersecurity and Privacy*

MDPI

Article

LDSEGoV: An Efficient Lightweight Digital-Signature Algorithm Based on CDLP and Provable Security for E-Governance Authentication

Seema Sirpal [1,2], Pardeep Singh [1,3] and Om Pal [1,3,*]

[1] Department of Computer Science, University of Delhi, Delhi 110007, India; ssirpal@ducc.du.ac.in (S.S.); psingh@cs.du.ac.in (P.S.)
[2] Delhi University Computer Centre, University of Delhi, Delhi 110007, India
[3] SafeQbit Technologies Private Limited, Noida 201318, India
* Correspondence: opal@cs.du.ac.in

Abstract

Digital signatures serve as a crucial cryptographic primitive in an e-governance system for authenticating citizen-government interactions. Traditional methods (DSA, ECDSA) impose computational overhead on resource-limited endpoints and centralized verification servers. While complex-number cryptography provides theoretical efficiency through the Complex Discrete-Logarithm Problem (CDLP), prior works often fail to meet the requirements for real-world applications. This paper advances the knowledge in lightweight cryptography by introducing LDSEGoV, a lightweight digital signature scheme for e-governance infrastructure. The proposed method overcomes the shortcomings of previous methods by incorporating sound modular arithmetic for consistent verification, using NIST-approved hash functions. Furthermore, we provide a comprehensive security analysis, including formal proofs of existential unforgeability (EUF-CMA) for the proposed scheme in the Random Oracle Model. Additionally, the experimental results show a 6.5× improvement in signing performance and a 24.76× improvement in verification performance over ECDSA, with a 61% reduction in signature size. These results demonstrate computational efficiency suitable for e-governance authentication scenarios.

Keywords: lightweight cryptography; e-governance; authentication; digital signatures; Discrete-Logarithm Problem (DLP); Random Oracle Model (ROM)

1. Introduction

Digital governance systems are transforming the interface between citizens and government through electronic service delivery, enabling scalable authentication for transactions such as tax returns, social security benefits, certificate issuance, and regulatory compliance. As Common Service Centers (CSCs), mobile portals, and verification systems are being set up by governments across the world to cater to millions of citizens, it becomes imperative for the authorities to ensure the authenticity, integrity, and non-repudiation of digitally signed documents. Digital signatures are an essential part of the security infrastructure, tying transactions to trusted signers and allowing for scalable verification without sacrificing efficiency. However, the computational cost of traditional signature schemes has proven to be a performance limitation for resource-constrained nodes such as CSCs, smart cards, and mobile devices, as well as for verification servers handling thousands of concurrent verification requests.

Check for updates

Received: 13 February 2026
Revised: 2 April 2026
Accepted: 30 April 2026
Published: 5 May 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

Cryptographic primitives such as RSA, DSA, and ECDSA are well-established and offer high security but incur significant computational overhead, making it difficult to scale up large electronic systems [1]. The time taken for ECDSA signing is around 0.699 ms, whereas for secure smart cards it is 150–200 ms. Verification, which takes about 3.368 ms/signature, limits throughput on single-server systems to about 297 signatures per second, which is insufficient to handle peak government service traffic. In addition, the DSA and ECDSA signatures consume 832 bits, including encoding overheads, resulting in high bandwidth requirements in rural e-governance networks that support constrained 2G/3G communication channels. Therefore, efficient cryptographic algorithms are required to facilitate e-governance infrastructure, enabling a trade-off between high security and low computational and communication overhead.

Complex number cryptography, which relies on the Complex Discrete-Logarithm Problem (CDLP), has extended the use case into the Gaussian integers $\mathbb{Z}_p[i]$ with the potential for theoretical efficiency improvements due to the expanded problem space, potential reduction in signature size, and improved arithmetic. While index calculus adaptations to CDLP face increased complexity resulting from the factorization of two-dimensional elements [2], we adopt a conservative security position. Our 2048-bit parameters comply with NIST recommendations for classical DLP [3] and exceed the recent DLP record of 795 bits [4] by 2.6×. Therefore, LDSEGoV ensures security at least as strong as classical DLP-based constructions currently used in governmental infrastructures (e.g., DSA, ECDSA), with the two-dimensional algebraic structure providing potential additional security margin. Recent implementations [5–7] demonstrate CDLP's practical viability, we note that LDSEGoV is the first to achieve a formal security validation.

The Shortened Complex Digital-Signature Algorithm (SCDSA), proposed by Mughal et al. [8], is the first attempt to leverage CDLP for lightweight digital signatures. Our analysis shows that SCDSA exhibit several shortcomings that make it impractical for use: it does not work for production-quality parameters ($p \geq 2048$ bits) because of inconsistent complex modular arithmetic in the exponentiation step; it is based on the deprecated SHA-1 hash function, which is vulnerable to collision attacks and has been discouraged by NIST in favor of more secure alternatives [9,10] and lacks any formal security analysis in the context of EUF-CMA security games.

This work introduces LDSEGoV, a formally secure CDLP-based digital signature scheme that addresses the weaknesses of SCDSA while maintaining the efficiency required by e-governance applications. LDSEGoV is designed with verification consistency in mind by requiring modular reduction over $\mathbb{Z}_p[i]$ to guarantee proper functionality with all possible parameter sizes, including those relevant to deployment with $p \geq 2048$ bits. LDSEGoV is based on hash functions with NIST-validated security proofs, with SHA-256 as the default due to its well-known 128-bit security. Moreover, SHAKE128 and SHAKE256 are also examined to meet the requirements of applications that demand variable-length hashing. The choice of SHA-256 is in line with current cryptographic practices and is designed to be efficient within the standard e-governance signature size of 320 bits.

Experimental validation demonstrates the verification rate of about 7350 signatures per second in single-threaded mode (1000 ms/0.136 ms per signature). This corresponds to a 24.76× speedup over ECDSA, which performs 297 signatures per second on the same hardware, demonstrating the ability to handle multiple authentication requests from hundreds of CSCs without horizontal scaling. The smaller size of the signature also helps address the bandwidth bottleneck common in rural 2G/3G networks.

In addition, the paper provides a comprehensive security analysis in both formal and informal ways. Formal security proof in the Random Oracle Model (ROM) for the proposed scheme LDSEGoV proves the proposed scheme to be Existential Unforgeability

under Adaptive Chosen-Message Attack (EUF-CMA), thus making the proposed scheme secure against forgery and replay attacks. The informal security proof demonstrates that the proposed scheme withstands various attacks.

1.1. Key Contributions

The primary contributions of this study are as follows:

1. Lightweight Digital Signature Scheme for E- Governance: This study introduces LDSEGoV, a signature scheme for CDLP specifically designed for digital governance scenarios that addresses the challenges of computation on resource-constrained endpoints and verification servers.
2. Corrected and Validated CDLP Implementation: Our cryptanalysis reveals significant limitations in a study by Mughal et al. [8] (SCDSA), such as verification failure for large parameters and the use of outdated SHA-1. We provide a corrected algorithm that ensures universal verification success for all parameter sizes, as validated by 1000 iterations with 2048-bit primes.
3. Formal and Informal Security Analysis: This work presents the first formally proven EUF-CMA secure scheme, with security established in the Random Oracle Model via a game-based reduction to the hardness of CDLP via the forking lemma, and shows resistance to forgery, replay, man-in-the-middle, and nonce-replay attacks augmented by an informal security analysis.
4. Comprehensive Performance Analysis: Comparison with DSA, ECDSA, and Mughal et al. [8] shows a speedup of 6.5 times in signing, a speedup of 24.76 times in verification, and a 61% reduction in signature size.
5. E-Governance Deployment Impact Analysis: We analyze the scalability of LDSEGoV in the CSC infrastructure and show that the performance of verification demonstrates computational capacity to handle 10–100 times traffic scaling, unlike ECDSA, which requires distributed clusters.

1.2. Paper Organization

The remainder of the paper is organized as follows. Section 2 introduces the cryptographic preliminaries and mathematical concepts related to complex modular arithmetic. In Section 3, we present a review of the existing literature and highlight the research gap. Section 4 describes the system and the problem formulation. The proposed LDSEGoV algorithm is described in Section 5. In Section 6, we detail security in the ROM and other security considerations. Performance comparison with other methods and implementation results are given in Section 7. Section 8 provides the conclusions and outlines future work.

2. Preliminaries

This section introduces the key concepts required to understand the cryptographic basis of the proposed lightweight digital signature method.

2.1. Discrete-Logarithm Problem (DLP) and Integer Factorization

The Discrete-Logarithm Problem (DLP) is a major field of study in cryptography, playing a pivotal role in the security designs underpinning many public-key cryptographic systems. Within the context of the DLP, an individual is provided with a generator α alongside an element β [11]. The objective is to find an integer x that satisfies the equation $alpha^x \equiv \beta \pmod{p}$, with p being a prime number. The security of most cryptographic schemes, especially DSA and ECDSA, depends on problems arising from the DLP. The security of digital signature schemes is primarily based on the difficulty of breaking the DLP. This complexity significantly restricts an adversary's ability to derive the secret key

from the public key. Another fundamental cryptographic problem is integer factorization. Given a large composite number N, the aim is to factor it into its prime components. Though DLP and Integer Factorization are separate issues, they both depend on solving number-theoretic problems to provide security, and thus resemble one another. The related CDLP in our proposed scheme, LDSEGoV, provides additional security by extending DLP into the complex-number domain.

2.2. Modular Arithmetic and Complex Numbers

Modern cryptography depends on the fundamental function known as modular arithmetic; its use on complex numbers is crucial to our proposed LDSEGoV. LDSEGoV uses modular arithmetic in the complex domain, which adds greater complexity and security even though conventional cryptographic methods operate in integer fields. LDSEGoV operates the modulus operation on both the real and imaginary components of complex numbers. The modulus of a complex number $z = a + bi$, where a and b are real numbers, is found by using the modulo operation on both a and b with respect to a large prime n. Adversaries have to negotiate two-dimensional number spaces, so this extension of complex numbers makes cryptanalysis more difficult in relation to traditional integer-based cryptosystems. Extending the scope to include complex numbers in cryptographic systems increases both security and computational complexity, thereby strengthening the system against cryptographic attacks.

2.3. Complex Discrete-Logarithm Problem (CDLP)

The central cryptographic challenge in LDSEGoV is CDLP, which is an extension of the Discrete-Logarithm Problem defined over complex numbers. Given a generator α and a complex number β, the CDLP is defined as finding an integer x such that $\alpha^x \equiv \beta \pmod{p}$, where p is a large prime number, and α and β are complex numbers [12]. Solving CDLP is computationally more challenging, thereby providing a strong basis for LDSEGoV security. The security of LDSEGoV is enhanced by the computational complexity of solving the DLP in the complex number domain. Although DLP-based schemes are vulnerable to attacks by sophisticated algorithms, CDLP offers greater security.

2.4. Digital-Signature Algorithm (DSA)

Among the most often used cryptographic techniques for creating and validating digital signatures is DSA. Based on DLP, DSA runs using the following procedures:

1. Key Generation: Select random $x \in \{1, 2, \ldots, \omega - 1\}$, compute $Y = \alpha^x \bmod p$, return (x, Y).
2. Signature Generation: Select random $\varepsilon \in \{1, 2, \ldots, \omega - 1\}$, $c = (\alpha^\varepsilon \bmod p) \bmod \omega$, $z = \varepsilon^{-1}(\mathcal{H}(M) + x \cdot c) \bmod \omega$, return (c, z).
3. Signature Verification: Compute $w = z^{-1} \bmod \omega$, $u_1 = \mathcal{H}(M) \cdot w \bmod \omega$, $u_2 = c \cdot w \bmod \omega$, $v = (\alpha^{u_1} \cdot Y^{u_2} \bmod p) \bmod \omega$, Verify whether $v = c$. If so, accept; else, reject.

Although DSA provides a strong security guarantee based on the DLP, it is still vulnerable to certain attacks, especially if the random value ε is chosen insecurely. The LDSEGoV improves the verification step and uses more robust cryptographic primitives to overcome these weaknesses.

2.5. Elliptic Curve Digital-Signature Algorithm (ECDSA)

ECDSA improves computational efficiency while providing the same level of security as DSA with smaller key sizes. Although ECDSA uses elliptic curve mathematics to generate key and signature pairs, the same process is followed as in DSA.

1. Key Generation: Select random $x \in \{1, 2, \ldots, \omega - 1\}$, compute $Y = x \cdot P$, return (x, Y).
2. Signature Generation: Select random $\varepsilon \in \{1, 2, \ldots, \omega - 1\}$, $c = (\varepsilon \cdot P)_x \bmod \omega$, $z = \varepsilon^{-1}(\mathcal{H}(M) + x \cdot c) \bmod \omega$, return (c, z).
3. Signature Verification: Compute $w = z^{-1} \bmod \omega$, $u_1 = \mathcal{H}(M) \cdot w \bmod \omega$, $u_2 = c \cdot w \bmod \omega$, $V = u_1 \cdot P + u_2 \cdot Y$, $v = V_x \bmod \omega$, verify whether $v = c$; if so, accept; else, reject.

ECDSA provides a more efficient alternative to DSA, especially in environments where smaller key sizes are advantageous. However, like DSA, it is vulnerable to attacks if the random value ε is reused or poorly chosen. LDSEGoV improves on these schemes by introducing complex-number-based cryptography to enhance overall security.

CDLP Correctness: For a valid signature (c, z):

$$\begin{aligned} V &= u_1 \cdot P + u_2 \cdot Y \\ &= \mathcal{H}(M) \cdot z^{-1} \cdot P + c \cdot z^{-1} \cdot (x \cdot P) \\ &= z^{-1}(\mathcal{H}(M) + x \cdot c) \cdot P \\ &= \varepsilon \cdot P \quad (\text{since } z = \varepsilon^{-1}(\mathcal{H}(M) + x \cdot c)) \end{aligned}$$

Therefore, $v = V_x = (\varepsilon \cdot P)_x = c$.

3. Related Work and Research Gap Analysis

The e-governance infrastructure has developed into three major domains in terms of digital signature schemes: the traditional schemes based on the integer factorization problem (RSA), the discrete logarithm-based schemes (DSA, ECDSA), and the lightweight schemes for resource-constrained environments, as well as the hybrid schemes combining both primitives with document verification primitives. This section covers representative works on the aforementioned topics, common security and performance limitations, and the clear positioning of the proposed LDSEGoV relative to recent research.

3.1. Classical Digital Signature Schemes for E-Governance

Sharif et al. [13] proposed a web-based PDF authentication system that uses RSA signatures and SHA-3 hashing. The study showed negligible processing costs across different PDF sizes. However, the proposed technique lacks content-aware verification beyond basic file hashing. In another study, Aufa et al. [14] analyzed the combination of RSA-1024 and DSA-512 for secure government communications and found that lower computational costs are appropriate for e-governance applications. This study focused on cryptographic efficiency but did not consider scalability issues in high-speed verification scenarios; in particular, single-server ECDSA signing supports only 297 signatures per second, which is inadequate for a national-level authentication system handling thousands of simultaneous Common Service Centers.

3.2. Lightweight Digital Signature Scheme

Given the computational limitations inherent to classical schemes, there has been a need for lightweight solutions better suited to resource-limited settings. Meshram et al. [15] designed a discrete logarithm scheme that leverages subtree topologies to minimize communication costs. Although they showed that the scheme is more efficient, it is not proven to be secure in the EUF-CMA model. It is also vulnerable to side-channel attacks and fault injection, which are undesirable properties for a government authentication infrastructure that must also be legally defensible and physically secure. Ahmed and Barukab [16] proposed a method that combines lightweight encryption with digital signatures for IoT, resulting in smaller signature sizes. However, their method fails to resist nonce-reuse

attacks. Yavuz et al. [17] focused on medical IoT authentication using scalable frameworks that emphasize device-level efficiency; however, their approach to healthcare telemetry applications does not apply to e-governance document authentication requirements, which require legally binding citizen transactions.

3.3. Complex Number-Based Cryptography and CDLP

Complex number cryptography relies on the Complex Discrete-Logarithm Problem (CDLP) to achieve efficiency through calculations in the Gaussian integer domain, represented by $\mathbb{Z}_p[i]$. Lalem et al. [18] revisited complex-number-based signatures for their efficiency in lightweight applications. Nevertheless, like previous studies, they did not address the issue of verification ambiguity, especially when using large parameter sizes, nor did they provide a proof of security. The study by Aung and Hla [7] explored the application of elliptic curve cryptography in relation to finite field theory and complex number systems. The study found that complex number theory could reduce the operational costs of elliptic curve cryptography. However, this study did not provide formal security analysis results for the application of complex number theory in elliptic curve cryptography. A study by Hla et al. [6] on intelligent computing methodologies for cryptography based on complex number theory emphasized the superiority of CDLP over DLP in terms of the problem space's dimensionality. However, the study did not provide a comparison of CDLP's robustness to DLP using conventional attack methodologies. The most recent study by Kamal et al. [5] proposed a conic curve-based approach for encrypting and digitally signing messages using complex number theory. Although this study showed that the proposed theory was computationally efficient, it did not provide a formal security analysis using traditional EUF-CMA models. Furthermore, this study did not provide results indicating that the proposed theory was verified for large parameter values.

Parallel advances in general DLP research have implications for CDLP security analyses. Sarkar et al. [19] presented a detailed survey of the computational hardness of the DLP, including index calculus and number field sieve methods in a prime field. The paper reinforces the safety margins and confirms parameters for 2048-bit primes. While recent work on hard instances of the DLP by Battarbee et al. [20] indicates that advances in cryptanalytic techniques are being made, no results have come close to threatening the security of any parameterized constructions of the DLP or the CDLP. The advances in DLP cryptanalysis by Barbulescu [21], including factorization and discrete logarithm algorithms, have implications for the choice of security parameters. These advances demonstrate the asymptotic sub-exponential time complexity of DLP and discrete logarithm computations, including the tower-NFS and FFS variants. The computation of a 795-bit discrete logarithm in a prime field by Boudot et al. [4] indicates the empirical security margins of CDLP, which is 2.6 times larger.

Cryptanalysis of SCDSA: Mughal et al. [8] proposed the Shortened Complex Digital-Signature Algorithm (SCDSA) for human-centric resource-constrained IoT networks, utilizing complex exponentiation to achieve smaller signatures. While the idea is very promising, our analysis of the implementation reveals some very important shortcomings, as follows:

1. Verification Failures with Production Parameters:The verification algorithm of SCDSA shows deterministic failures for the prime modulus p, which meets the condition $p \geq 2048$ bits (required for 128-bit security). The main reason is the lack of modular reduction in the complex exponentiation $(a + bi)^k \bmod p$. The original scheme reduces the result only after complex exponentiation, which causes integer overflow and incorrect signature rejection. The fixed code ensures modular reduction is performed after each multiplication operation in the square and multiply method to ensure consistent verification results for all possible parameter sizes.

2. Deprecated Cryptographic Primitives: SCDSA employs SHA1 hashes, which NIST deprecated in 2011 due to collision attacks [9,22]. New e-governance applications require NIST-approved alternatives like SHA-256, SHA-3, and SHAKE-256 with an approximate collision resistance of about $2^1 28$.
3. Lack of Formal Security Validation: SCDSA does not provide any formal security analysis and proof security which is a concern for legally binding government authentication.
4. Ambiguous Modular Arithmetic Semantics: The division is defined as $s = k/(r + U_a) \bmod q$, which is a combination of division and modular inversion. This definition creates ambiguity. It is defined in terms of the calculation of the modular multiplicative inverse as $s = k \cdot (r + U_a)^{-1} \bmod q$. The modular multiplicative inverse of $(r + U_a)^{-1}$ can be calculated using the Extended Euclidean Algorithm.

3.4. Cryptanalytic Advances Against Classical Schemes

Recent cryptanalytic work has revealed weaknesses in widely used signature schemes. The Minerva attack [23] demonstrated the recovery of ECDSA private keys via electromagnetic side-channel analysis of partial nonce leakage, thereby undermining confidence in standard discrete logarithm-based schemes. Macchetti [24] proposed related-nonce attacks that allow full ECDSA key recovery with poor randomness, requiring only a few observations of signatures.

Fault injection attacks remain a threat to robust cryptographic implementations. Cao et al. [25] have shown that lattice-based fault attacks can recover ECDSA private keys based on weak curves via single fault injection attacks. Hou et al. [26] have provided a thorough survey of the vulnerabilities of RSA, DSA, and ECDSA to fault-injection attacks in commercial implementations, highlighting the gap between theoretical security proofs and physical attacks.

Empirical data on the feasibility of the Bleichenbacher-style RSA signature forgery attack is presented by Blessing et al. [27]. Timing side-channel attacks on Infineon's hardware implementation of ECDSA are shown by Geimer [28]. These results make it clear that cryptographic protocols require not only formal security proofs but also constant-time and fault-tolerant implementations to secure the government authentication infrastructure.

4. Identification of Problem and System Modeling

4.1. Problem Identification

The commonly used signature schemes, such as DSA, are based on the DLP. The three basic public parameters are the large prime p, the small prime ω such that ω divides $p - 1$, and the generator parameter α, which is defined as $\alpha = h^{(p-1)/\omega} \bmod p$. This model imposes significant computational complexity, particularly on resource-limited e-governance devices such as CSC terminals, smart cards, and mobile devices in rural areas with limited power and processing capacity. While solving this, researchers opt for smaller instances of p and ω. Reducing the parameter sizes (i.e., smaller p and ω) may ease the computational burden to some extent; yet, this approach provides an unacceptably poor security-performance trade-off. For instance, reducing p from 2048 bits to 1024 bits cuts signing time by around 75%, simultaneously reducing the level of security from 112-bit to 80-bit, which is not strong enough for government authentication schemes requiring long-term validity of signatures (7–10 years). On the other hand, keeping appropriate security levels with regular parameters($p \geq 2048$ bits, $\omega \geq 256$ bits) raises a dual bottleneck: The signing speed of ECDSA on smartcards worsens to around 150–200 ms. In comparison, the verification process at 3.368 ms per signature limits the throughput of single-server systems to 297 signatures per second, which is not sufficient for maximum e-governance traffic rates

of over thousands of simultaneous submissions. Additionally, the size of DSA/ECDSA signatures is 832 bits, which is equivalent to 416 bytes for a four-document citizen application when communicated through bandwidth-limited rural networks (64–384 kbps). These challenges, further exacerbated by the increased use of digital services in the context of digital governance, require the development of cryptographic algorithms tailored to the specific constraint profile of the e-governance domain, characterized by high transaction volumes, diverse device capabilities, and intermittent connectivity.

While the complex-number cryptography offers computational advantage, the previous implementations (SCDSA [8]) have been found to have several important shortcomings that make them unsuitable for use in legally binding e-governance applications, where the validity of signatures needs to resist cryptanalytic attacks over an archival period of seven to ten years: (1) irregular modular arithmetic operations that result in verification errors for parameters p with 2048 bits or larger; (2) use of the SHA-1 hash function, which is no longer recommended due to the risk of collision attacks, and (3) lack of rigorous security proofs for resistance to adaptive chosen-message attacks.

This analysis reveals a gap in the signature schemes used, as they compromise security or increase computation cost in e-governance systems. Previous signature schemes have faced challenges in terms of implementation and validation. Thus, there is a need for an efficient and secure CDLP signature scheme for scalable authentication in government applications.

4.2. System Model for LDSEGoV

LDSEGoV introduces lightweight e-governance for constrained endpoints, such as CSC terminals, smart cards, and mobile apps. Each entity in the system, the citizen, the CSC operator, and the government department, has a pair of keys: the private key x remains securely on the device in the HSM for servers or the secure enclave in the smart cards, and the public key Y is published in the government PKI directory. As depicted in Figure 1, upon a citizen's request for the service through the CSC, the terminal will sign the application using the private key x. The government verification server will then verify the signature using the public key Y.

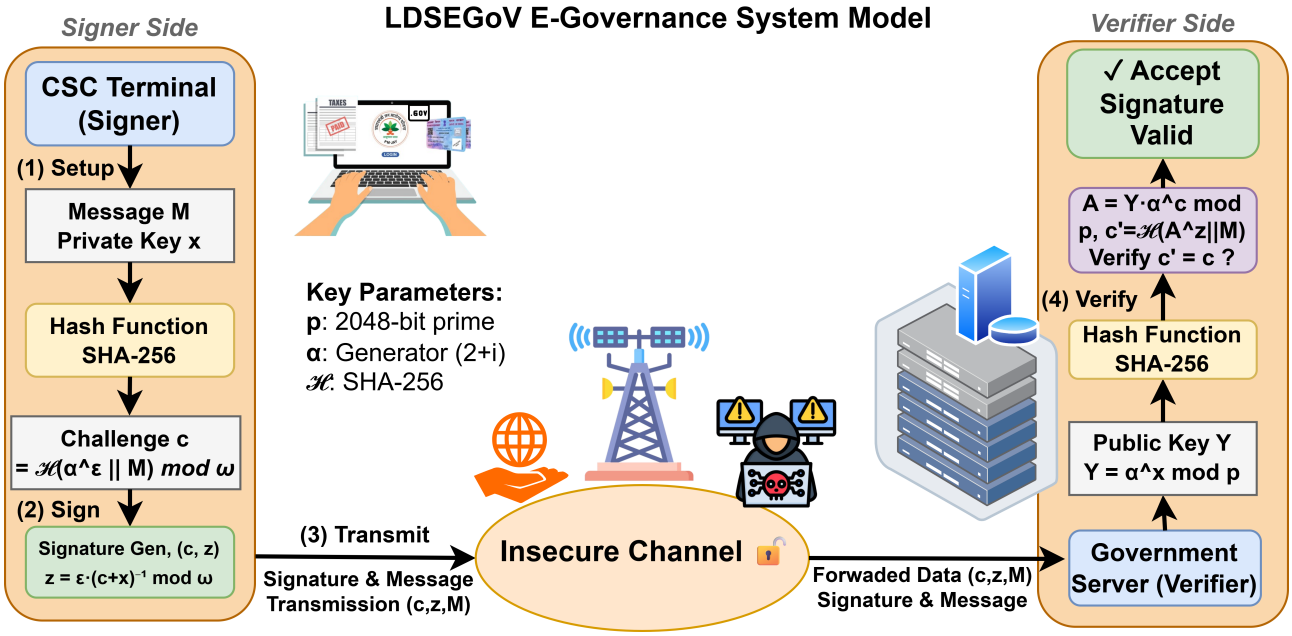


Figure 1. LDSEGoV system model illustrating the complete authentication workflow across four phases: (1) system setup and key generation at the CSC terminal; (2) signature generation; (3) signature and message transmission (c, z, M) over insecure public network; and (4) server-side verification.

The process of generating the signature involves the signer (CSC terminal) calculating the pair (c, z) using the ephemeral nonce ε and the private key x, followed by the transmission of the signature through an insecure channel. The verifier (government server) then checks $c' = c$ using the public key Y, with a new nonce ε for every signature to prevent replay attacks.

4.3. Threat Model for LDSEGoV

We analyze an adversary $\mathcal{A}$ with network control (capable of intercepting and modifying messages), polynomial-time computational resources (in particular, unable to solve the Computational Diffie–Hellman Problem for modulus $p \geq 2048$ bits), and adaptive oracle access (capable of observing legitimate CSC signatures). The LDSEGoV defenses against the most important attack vectors are the following.

Adversary Capabilities:

1. *Network Control:* Complete access to communication channels enables the interception, modification, and insertion of messages exchanged between citizen service centers (CSCs) and government servers on the assumption of an insecure channel.
2. *Computational Power:* A polynomial-time bounded adversary is assumed capable of performing up to 2^{80} operations but is unable to solve Discrete-Logarithm Problem (CDLP) instances with modulus $p \geq 2048$ bits within a feasible time.
3. *Oracle Access:* Adaptive chosen-message queries to signing oracles are considered, enabling observation of legitimate CSC transactions and reflecting real-world scenarios in which attackers observe citizen application submissions.
4. *Insider Threats:* The potential compromise of CSC operator credentials or temporary access to unattended terminals is acknowledged, while direct access to private keys stored within hardware security modules or secure enclaves is explicitly ruled out.

Threat Scenarios and Defense:

- *Man-in-the-Middle:* The modification of the signed message M to M' is prevented since $c = \mathcal{H}(\alpha^{\varepsilon} \parallel M)$ binds signature to message hash through the property of 2^{128} preimage resistance of SHA-256.
- *Replay Attacks:* Using a new nonce value ε for each signature prevents the same messages from being signed with the same pair (c, z). Including timestamps in the signed data in e-governance systems offers a further safeguard against repeated claims for welfare benefits.
  Using a fresh nonce ε for every signature avoids signatures on the same message with the same pair (c, z). For e-governance applications, adding timestamps to the signed data would help avoid multiple identity claims for welfare benefits.
- *Forgery:* The generation of valid (c^*, z^*) without knowledge of the private key x is equivalent to the CDLP problem in the cyclic group, which is shown to be EUF-CMA secure in Theorem 1.
- *Eavesdropping:* The private key x is never transmitted. The response $z = \varepsilon \cdot (c + x)^{-1} \bmod \omega$ hides both x and ε using one-way modular arithmetic.
- *Nonce Reuse:* The ability to see two signatures with the same value of ε allows key extraction using the forking lemma. This is addressed by requiring fresh randomness and by using hardware random number generators (e.g., TPM 2.0, secure enclaves).
- *CSC Compromise:* Hardware-protected storage solutions (e.g., JavaCard, cloud-HSM) protect against private key extraction even when terminal access is available.

5. Proposed Lightweight Digital Signature Scheme

This section presents our proposed scheme, LDSEGoV: Lightweight Digital Signature for E-Governance, a corrected and formally validated complex-number signature scheme.

The scheme fits within the proposed four-algorithmic framework. A pseudocode for the proposed method is described in Algorithms 1–4, while the full algorithm specification consists of four algorithms: system parameter setup (Algorithm 5), key pair generation (Algorithm 6), signing (Algorithm 7), and verification (Algorithm 8). The notations and parameters used in LDSEGoV are enumerated in Table 1.

Table 1. Notation and parameters used in LDSEGoV.

Symbol	Description
p	Large prime modulus ($\|p\| \geq 2048$ bits)
ω	Subgroup order, where $\omega = p - 1$
α	Complex generator in $\mathbb{Z}_p[i]$ with order ω
ε	Ephemeral random nonce, $1 < \varepsilon < \omega$
$\sigma = (c, z)$	Digital signature pair
$\mathcal{H}(\cdot)$	Cryptographic hash function (SHA-256/SHAKE256)
Y	Public verification key (complex number mod p)
x	Private signing key, $1 < x < \omega$
M	Message to be authenticated (End User's data (i.e., a PDF document, file, application payload, email, or any digital information needing authenticity verification)
c	Challenge: $c = \mathcal{H}(\alpha^{\varepsilon} \parallel M) \bmod \omega$
z	Response: $z = \varepsilon \cdot (c + x)^{-1} \bmod \omega$
c'	Recomputed challenge during verification

5.1. Pseudocode for LDSEGoV

The proposed LDSEGoV protocol is described by Algorithms 1–4. The provided pseudocode gives an insight into the process of initializing, generating keys, signing messages, and verifying signatures in a structured manner. Algorithm 1 initializes the system by generating the parameters (p, ω, α) according to the given security parameter. The private-public key pair generation algorithm is shown in Algorithm 2, where the private key x is chosen and the public key $Y = \alpha^x \bmod p$ is computed. Signature generation uses a random nonce and the challenge-response method, as shown in Algorithm 3. Algorithm 4 involves the reconstruction of the challenge and verification of its correctness.

Algorithm 1 System Setup

Require: Security parameter λ
Ensure: System parameters (p, ω, α)
1: Choose a large prime $p > 2^{\lambda}$, $\omega \leftarrow p - 1$, $\alpha \leftarrow$ generator (complex number in $\mathbb{Z}_p[i]$), return (p, ω, α)

Algorithm 2 Key Generation

Require: System parameters (p, ω, α)
Ensure: Key pair (x, Y)
1: Pick a random $x \in \{1, 2, \ldots, \omega - 1\}$, calculate $Y = \alpha^x \bmod p$, return (x, Y)

Algorithm 3 Signature Generation

Require: Message M, private key x, system parameters (p, ω, α)
Ensure: Signature $\sigma = (c, z)$
1: Select a random number $\varepsilon \in \{1, 2, \ldots, \omega - 1\}$, $c = \mathcal{H}(\alpha^{\varepsilon} \parallel M) \bmod \omega$, $z = \varepsilon \cdot (c + x)^{-1} \bmod \omega$, return (c, z)

Algorithm 4 Signature Verification

Require: Message M, signature (c, z), public key Y, system parameters (p, ω, α)
Ensure: Decision $\{\text{accept}, \text{reject}\}$
1: Calculate α^c, find $A = Y \cdot \alpha^c \bmod p$, $c' = \mathcal{H}(A^z \parallel M) \bmod \omega$, if $c' = c$ then accept; otherwise, reject.

5.2. Overview of the Proposed Method: LDSEGoV

The LDSEGoV scheme operates in the Gaussian integer ring $\mathbb{Z}_p[i]$, leveraging complex modular arithmetic to achieve computational efficiency and maintain cryptographic security. Unlike conventional signature schemes that operate over prime fields or elliptic curve groups, the LDSEGoV scheme extends the DLP into the two-dimensional complex plane, where both the real and imaginary parts contribute to the difficulty of key recovery. A complete algorithmic specification consists of four algorithms: system parameter setup (Algorithm 5), key pair generation (Algorithm 6), signing (Algorithm 7), and verification (Algorithm 8), as follows.

Algorithm 5 LDSEGoV System Setup

Require: Security parameter λ
Ensure: System parameters (p, ω, α)
1: Choose a large prime $p > 2^\lambda$
2: Set group order $\omega \leftarrow p - 1$
3: Select generator $\alpha = (\alpha_r, \alpha_i)$ where $\alpha_r, \alpha_i \in \mathbb{Z}_p^*$
4: **return** (p, ω, α)

Algorithm 6 Key Generation

Require: System parameters (p, ω, α)
Ensure: Key pair (x, Y)
1: **repeat**
2: $x \xleftarrow{\$} \{1, 2, \ldots, \omega - 1\}$
3: **until** $\gcd(x, \omega) = 1$
4: $Y \leftarrow \alpha^x \bmod p$ {Complex exponentiation}
5: **return** Private key x, Public key Y

Algorithm 7 Signature Generation

Require: Message M, private key x, parameters (p, ω, α)
Ensure: Signature $\sigma = (c, z)$
1: *attempts* $\leftarrow 0$
2: **while** *attempts* $<$ *MAX_ATTEMPTS* **do**
3: **repeat**
4: $\varepsilon \xleftarrow{\$} \{1, 2, \ldots, \omega - 1\}$
5: **until** $\gcd(\varepsilon, \omega) = 1$
6: $(u, v) \leftarrow \alpha^\varepsilon \bmod p$ {Extract real, imaginary parts}
7: $c \leftarrow \mathcal{H}(u \parallel v \parallel M) \bmod \omega$
8: **if** $\gcd(c + x, \omega) \neq 1$ **then**
9: *attempts* $\leftarrow$ *attempts* $+ 1$
10: **continue**
11: **end if**
12: $z \leftarrow \varepsilon \cdot (c + x)^{-1} \bmod \omega$
13: **return** $\sigma = (c, z)$
14: **end while**
15: **abort** with error "Signature generation failed"

Algorithm 8 Signature Verification

Require: Message M, signature $\sigma = (c, z)$, public key Y, parameters (p, ω, α)
Ensure: Decision $\{\text{accept}, \text{reject}\}$
if $c \notin [1, \omega - 1]$ **or** $z \notin [1, \omega - 1]$ **then**
  return reject
end if
$\alpha^c \leftarrow (\alpha_r + i\alpha_i)^c \bmod p$ {Complex exponentiation}
$A \leftarrow Y \cdot \alpha^c \bmod p$ {Complex multiplication}
$(u', v') \leftarrow A^z \bmod p$ {Extract real, imaginary parts}
$c' \leftarrow \mathcal{H}(u' \parallel v' \parallel M) \bmod \omega$
if $c' = c$ **then**
  return accept
else
  return reject
end if

The message M (end user's data, i.e., file, PDF document, application payload, etc.) is signed by the signature generation mechanism based on the operation of complex exponentiation $\alpha^\varepsilon \bmod p$, where the base $\alpha = \alpha_r + i\alpha_i$ is a complex generator, and the exponent ε is a per-message random value. The result of the exponentiation is a commitment point $(u, v) \in \mathbb{Z}_p \times \mathbb{Z}_p$, which is used to link the signature to the message M and the ephemeral randomness. The challenge part c is derived from the hash function by hashing the commitment point together with the message: $c = \mathcal{H}(u \parallel v \parallel M) \bmod \omega$. The response part z is then calculated as $z = \varepsilon \cdot (c + x)^{-1} \bmod \omega$.

One of the basic security features of the LDSEGoV protocol is the mandatory use of new randomness ε in each signature. This ensures that even when the same message is signed on different occasions, the signatures (c, z) are different because of the unpredictability of α^ε. If the same randomness ε is reused, then an attacker can easily extract the private key x by algebraic manipulation of two different signatures with the same randomness ε but different challenges c_1, c_2.

Verification calculates the original commitment by computing $A = Y \cdot \alpha^c \bmod p$ and then raising it to the power of the response, resulting in $A^z \bmod p$. The following algebraic identity justifies the validity of this process:

$$(Y \cdot \alpha^c)^z = (\alpha^x \cdot \alpha^c)^{\varepsilon \cdot (c+x)^{-1}} = \alpha^{(x+c) \cdot \varepsilon \cdot (c+x)^{-1}} = \alpha^\varepsilon$$

Therefore, a valid signature will result in the verifier reconstructing the same commitment (u, v) that the signer computed originally, which further leads to $c' = c$ and acceptance of the signature.

5.3. Implementation Considerations

SHA-256 hash function is employed to ensure compliance with NIST FIPS 180-4 and to provide collision resistance. As alternatives for variable-length outputs, SHAKE-128 and SHAKE-256 can be used without any modifications. All complex arithmetic operations must use uniform modular reduction after each operation to guarantee deterministic computation and prevent verification failures. Key storage requires hardware security: USB tokens (PKCS#11) for CSC personnel, HSMs (FIPS 140-2 level 3) for government agencies, and platform-specific secure enclaves for mobile apps. Public keys Y are embedded in X.509 certificates (RFC 5280) with OID mappings tailored to our scheme.

5.4. Multi-Option Parameter Selection

In addition, the security of LDSEGoV is enhanced by adding the Multi-option Parameter-Selection (MPS) technique to the signature generation and verification process. To improve resistance against cryptanalytic attacks that rely on fixed algebraic structures, LDSEGoV proposes three different response formulas. Each of these formulas offers the same level of security in CDLP but employs different algebraic expressions involving the challenge c and the secret key x. As a result, it becomes much more challenging for an attacker to forge the signature as they need to account for all the alternatives. We have provided three distinct formulations for both the signature and verification processes, along with their respective proofs of correctness and consistency, in Table 2. The technique provides immunity against potential cryptographic attacks, such as forgery and signature impersonation. Deployment Recommendation: In the context of e-governance systems, MPS-1 is recommended as the default. MPS-2 and MPS-3 can be used in scenarios that demand greater algorithmic flexibility or better resistance to unknown structural attacks. The chosen variant must be included in the signature metadata to ensure the verifier uses the correct verification formula.

Table 2. Correctness proofs for multi-option signature and verification.

ID	Response z	Verification	Correctness
MPS-1	$\varepsilon/(c+x) \bmod \omega$	$A = Y \cdot \alpha^{c}$	$A^{z} = (\alpha^{x+c})^{\varepsilon/(c+x)} = \alpha^{\varepsilon}$
MPS-2	$\varepsilon/(x+c^{2}) \bmod \omega$	$A = Y \cdot \alpha^{c^{2}}$	$A^{z} = (\alpha^{x+c^{2}})^{\varepsilon/(x+c^{2})} = \alpha^{\varepsilon}$
MPS-3	$\varepsilon/(xc+1) \bmod \omega$	$A = Y^{c} \cdot \alpha$	$A^{z} = (\alpha^{xc+1})^{\varepsilon/(xc+1)} = \alpha^{\varepsilon}$

5.5. Example Illustrating the Working of LDSEGoV

(I) Complex Modular Arithmetic

Operations are performed in the ring $\mathbb{Z}_p[i]$, where i is the imaginary unit. For an instance, `complex_mod(`w, p`)` reduces the real and imaginary parts of w modulo p, and exponentiation α^{ε} mod p uses a square-and-multiply approach adapted for complex numbers. The use of complex numbers expands the problem space for the DLP and Computational Diffie–Hellman (CDH) problems. Solving DLP in $\mathbb{Z}_p[i]$ is conjectured to be harder than in integer fields due to its higher dimensionality (real and imaginary components) and non-commutative structures when combined with modular arithmetic.

(II) System Setup for LDSEGoV

The LDSEGoV algorithm is engineered with best parameters to guarantee secure and efficiency. The prime number 5,915,587,277 is used as modulus p, specifying the arithmetic between complex numbers to be in $\mathbb{Z}_p[i]$. The order of a subgroup is selected ω = 5,915,587,276, and it holds $\omega|(p-1)$, which leads to the existence of the cyclic subgroup in the Gaussian integer field. We chose α to be the complex number $11 + 12i$ which, when taken modulo p, becomes $4 + 5i$. All in-subgroup public-key and ephemeral computations are based on this generator. For the secret key we took $x = 5$ which is a uniformly random integer in $[1, \omega - 1]$, and thus invertible modulo ω. Similarly, the ephemeral key used during signature generation was set to $\varepsilon = 4$ to satisfy $\gcd(\varepsilon, \omega) = 1$ to prevent leakage of secret key material. Key generation involves computing the public key $Y = \alpha^{x} \bmod p$. Signature generation involves hashing the ephemeral exponentiated value along with the message and computing a signature (c, z). Verification recalculates the expected hash and compares it with c.

(III) Key Generation

Computing α^{5} mod 7 using square-and-multiply:

1. Binary of 5: `101`
2. $\alpha = 4 + 5i$
3. Step 1: $\alpha^1 = (4, 5)$
4. Step 2: $\alpha^2 = (4 + 5i)^2 = (4^2 - 5^2, 2 \cdot 4 \cdot 5) = (-9, 40) \Rightarrow (5, 5)$
5. Step 3: $\alpha^4 = (5 + 5i)^2 = (25 - 25, 50) = (0, 50) \Rightarrow (0, 1)$
6. Step 4: $\alpha^5 = g^4 \cdot g = (0 + i)(4 + 5i) = (-5 + 4i) \Rightarrow (2, 4)$

Alice's public key: $Y = (2, 4)$.

(IV) Generation of Signature

1. Compute ephemeral value: $\alpha^\varepsilon = (4 + 5i)^4 \Rightarrow \alpha^4 = (0, 1)$.
2. Hash the tuple with the message "hello":
   $c = \mathcal{H}(\alpha^\varepsilon \parallel \texttt{hello}) = \mathcal{H}(0 \parallel 1 \parallel \texttt{hello})_{\text{SHA256}} = 11 \Rightarrow c = 11 \bmod 3 = 2$.
3. Check invertibility: $\gcd(c + x, \omega) = \gcd(2 + 5, 3) = \gcd(7, 3) = 1$.
4. Compute inverse and z:
   $(c + x)^{-1} \bmod \omega = 7^{-1} \bmod 3 = 1$.
   $z = \varepsilon \cdot (c + x)^{-1} \bmod \omega = 4 \cdot 1 \bmod 3 = 1$.
5. Final Signature: $\sigma = (c, z) = (2, 1)$.

(V) Verification of Signature

1. Compute α^c: $\alpha^2 = (4 + 5i)^2 = (5, 5)$.
2. Multiply with public key:

$$\begin{aligned} Y \cdot \alpha^c &= (2 + 4i)(5 + 5i) = (2 \cdot 5 - 4 \cdot 5,\ 2 \cdot 5 + 4 \cdot 5) \\ &= (10 - 20,\ 10 + 20) = (-10,\ 30) \\ &\equiv (4,\ 2) \bmod 7. \end{aligned}$$

3. Raise to z: $\alpha'^\varepsilon = (4 + 2i)^1 = (4, 2)$.
4. Recompute hash: $c' = \mathcal{H}(4 \parallel 2 \parallel \texttt{hello}) = 2$.
5. Compare: $c' = c \Rightarrow$ Signature is VALID.

6. Security Analysis of LDSEGoV Scheme

6.1. Formal Security Analysis in Random Oracle Model (ROM)

We prove the security of LDSEGoV through a formal game proof in the Random Oracle Model (ROM). In this model, the hash function $\mathcal{H}$ is modeled as an ideal "black-box" that returns uniformly random values for every different input. This simplification helps in making formal security proofs and captures the main characteristics of cryptographic hash functions, such as SHA-256.

Theorem 1. *If the Complex Discrete-Logarithm Problem (CDLP) is computationally intractable, then LDSEGoV is existentially unforgeable under adaptive chosen-message attacks (EUF-CMA) in the Random Oracle Model.*

Proof of Theorem 1. This theorem is proved by a series of four games, starting from the standard definition of EUF-CMA security and gradually changing the environment until a solution to a CDLP problem can be obtained. Each game differs slightly from the previous one, allowing a limit to be placed on the adversary's advantage. □

Proof Sketch. We provide a reduction showing that having an efficient forger implies having a solver for CDLP, and this is achieved using the forking lemma method. The formal proof is provided in Appendix A. □

6.2. Informal Security Analysis

In addition to the formal security proof in the ROM, we analyze LDSEGoV's resistance to real-world cryptographic attacks that may arise in the context of e-governance.

1. Resistance to Key Recovery Attacks: The private key x is randomly chosen from $\{1, 2, \ldots, \omega - 1\}$ ensuring $\gcd(x, \omega) = 1$, which creates a large key space and makes brute-force attacks infeasible. The public key $Y = \alpha^x \mod p$ is derived using complex exponentiation, making the reverse computation equivalent to solving CDLP, which is considered computationally hard.
2. Resistance to Forgery Attacks: Each signature uses a fresh random nonce ε, rendering repeated-signature correlation attacks ineffective. The hash function $\mathcal{H}(u \parallel v \parallel M)$ tightly couples the message to the signature, providing collision resistance and preventing message substitution.
3. Resistance to Replay Attacks: The uniqueness of the nonce ε in every valid signature (c, z) ensures that even if an attacker reuses a signature, the context mismatch can be detected by the verifier.
4. Resistance to Chosen-Message Attacks: The value $z = \varepsilon \cdot (c + x)^{-1} \mod \omega$ binds the signature to the private key. Without knowledge of x, it is computationally infeasible to create valid signatures for attacker-chosen messages.
5. Collision Resistance: The use of a secure cryptographic hash function ensures that it is infeasible to find two different message inputs producing the same $c = \mathcal{H}(u \parallel v \parallel M \mod \omega$, thereby ensuring message integrity.
6. Preimage Resistance: The private key x is never exposed and cannot be derived from the public signature elements, as the equation $z = \varepsilon \cdot (c + x)^{-1} \mod \omega$ hides both ε and x, making reverse engineering of x infeasible.
7. Second Preimage Resistance: The hash function $\mathcal{H}$ provides strong second preimage resistance, which prevents an adversary from finding a different message M' that produces the same hash value as M in the computation of c.
8. Resistance to Side-Channel Attacks: A fresh, random ε for each signature ensures that power or timing-analysis attacks are ineffective due to the lack of repeatable computation patterns.

Implementation Security

Government deployments must resist timing, power analysis, and fault injection attacks. Well-established countermeasures to side-channel attacks on Discrete-Logarithm Problem (DLP) signatures, such as constant-time exponentiation, exponent blinding, and hardware security modules, can be applied to LDSEGoV as well because of the complex arithmetic in $\mathbb{Z}_p[i]$ that is reduced to arithmetic in $\mathbb{Z}_p$. The same basic operations that remain susceptible to side-channel attacks in classical DLP systems make them equally susceptible to countermeasures. For instance, to compute $\alpha^x \mod p$ where $\alpha \in \mathbb{Z}_p[i]$, we use the same square and multiply techniques as in classical DLP systems and thus inherit the same vulnerabilities and countermeasures. Deployment in critical infrastructure should use Trusted Platform Modules (TPMs) (TPM 2.0) or processor-based secure environments to protect private keys, as recommended by current government standards for cryptographic implementation.

Summary: LDSEGoV is resistant to both theoretical attacks (as shown by formal verification) and practical attacks (achieved through careful nonce handling and constant-time arithmetic). The combination of CDLP security, cryptographic hash binding, and randomized signing provides a defense-in-depth strategy for the e-government authentication infrastructure.

7. Implementation and Result Analysis

7.1. Implementation Details

LDSEGoV was implemented using Python 3.10. The `gmpy2` library, version 2.1.5, was used for arbitrary-precision arithmetic and performing modular operations. Complex numbers were represented as tuples, and complex multiplication was performed using modular arithmetic. Modular exponentiation was performed using the square-and-multiply algorithm with left-to-right binary scanning. The hash functions were implemented using Python's built-in `hashlib` module, which implements SHA-256, meeting NIST's FIPS 180-4 specification. The prime numbers were generated using the `gmpy2.next_prime()` function with 2048-bit parameters. The generator used for the multiplicative group was set to $\alpha = 2 + i$. The random nonce was generated using Python's built-in `secrets.randbits()` function, which makes use of OS entropy sources (`/dev/urandom` on Linux. The code was not optimized at the assembly or hardware levels, so the results reflect the algorithms' efficiency. The performance comparison between DSA and ECDSA was carried out using Python's built-in `cryptography` module, with a version of 41.0.7, and default parameters, i.e., ECDSA over NIST's 256-bit curve and DSA with 2048-bit parameters. The implementation of SCDSA was made according to the specifications of Mughal et al. [8]. The arithmetic primitives used were the same. The code was run single-threaded on the same hardware.

7.2. Experimental Setup

We conducted experimental verification on commodity hardware: an Intel Core i7-10700 processor (8 cores, 2.90 GHz base frequency), 16 GB of DDR4 RAM, and Ubuntu Linux 22.04 LTS (5.15.0 kernel). All algorithms were executed 1000 times each with the timing captured using Python's `time.perf_counter()` function. We used the median value to help eliminate noise from operating system scheduling. We used the following parameter values: $p = 2048$ bits (prime modulus), $\omega = p - 1$ (group order), and message length $|M| = 256$ bits (output size of SHA-256 hash function). Results are shown as medians over 1000 iterations, with standard deviations $\sigma_{sign} = 0.008$ ms and $\sigma_{verify} = 0.012$ ms, indicating consistent performance, as the coefficient of variation is below 5%.

The performance and robustness of the proposed complex number-based Digital-Signature Algorithm, LDSEGoV, are evaluated in this section. The algorithm operates on complex integers modulo a large prime, leveraging their cryptographic robustness and algebraic complexity. To determine the efficiency of the LDSEGoV algorithm, a comparison test was conducted among the SCDSA, DSA, and ECDSA algorithms. Experimental verification carried out on commodity hardware shows a verification throughput of ≈7350 signatures per second in single-threaded mode (1000 ms/0.136 ms per signature), which is a 24.76× improvement over ECDSA, which reaches 297 signatures per second on the same hardware, sufficient to handle concurrent authentication requests from hundreds of CSCs without horizontal scaling. The performance of each algorithm was tested for 1000 iterations, and the results are presented in Table 3. Performance analysis was conducted based on three important functions: key generation, signature generation, and verification time, measured in milliseconds (ms), using the time.perf_counter() Python function.

Table 3. Performance comparison of digital-signature scheme mean execution time in milliseconds (ms) over 1000 iterations; measurements using Python's time.perf_counter().

Algorithm	KeyGen (ms)	Sign (ms)	Verify (ms)
LDSEGoV (Proposed)	0.0351	0.1080	0.1360
SCDSA	0.0359	0.0920	0.2980
DSA	602.8861	0.9266	0.5561
ECDSA	0.7814	0.6999	3.3681

The key results show that LDSEGoV achieves a consistent verification success rate across all hash functions, thereby establishing its effectiveness. In contrast, SCDSA showed unstable verification results, which LDSEGoV corrected. LDSEGoV outperforms DSA and ECDSA in key pair generation, signature generation, and verification with execution times of less than 0.13 ms. Although SCDSA showed faster signature generation at 0.092 ms, its high verification time of 0.298 ms and high failure rate make it unsuitable for practical e-governance applications.

7.3. Performance Analysis Across Varying Message Sizes

The performance of LDSEGoV, SCDSA, ECDSA, and DSA in generating signatures was evaluated for message sizes ranging from 1 MB to 10 MB. As shown in Figure 2a, LDSEGoV had the lowest signature generation time (0.07–0.14 ms), performing better than ECDSA (0.88–0.99 ms) and DSA (0.62–0.77 ms) for all message sizes. SCDSA showed marginally faster signature generation (0.07–0.13 ms), but its unreliable verification negates this advantage. ECDSA showed the largest fluctuation (±12%), suggesting it may be sensitive to message size; LDSEGoV maintained stability (±8%). The verification latency was also tested, and the results were statistically evaluated. As shown in Figure 2b, LDSEGoV achieved sub-millisecond verification (0.08–0.2 ms), 2.5 times faster than SCDSA (0.25–0.39 ms) and 15–20 times faster than ECDSA (3.09–4 ms). DSA (0.49–0.61 ms) performed competitively but was still 3 times slower than LDSEGoV. SCDSA's verification delays (0.25–0.39 ms) highlight its impracticality for real-world deployment. LDSEGoV and SCDSA performed similarly regarding communication overhead, as shown in Figure 2c. Moreover, LDSEGoV's low and stable latency makes it suitable for high-throughput applications. A 92% reduction in communication costs translates to lower energy consumption. Moreover, LDSEGoV's compatibility with SHA3 and SHAKE128/256 hash functions ensures alignment with NIST standards.

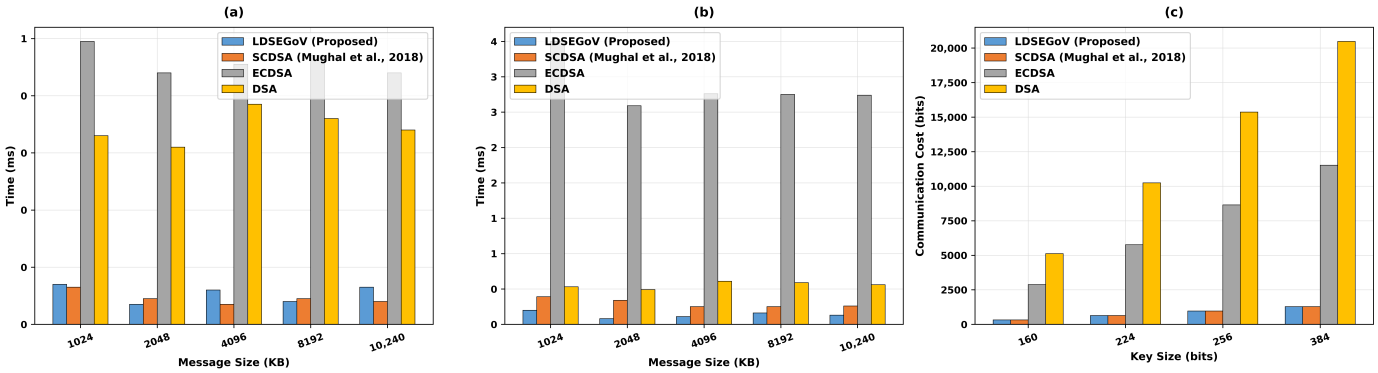


Figure 2. Comparative performance analysis of the proposed algorithm with RSA, ECDSA and Mughal et al. [8] (SCDSA) in terms of (**a**) signature-generation time in milliseconds (ms); (**b**) verification time in milliseconds (ms); (**c**) communication cost in bits over varying key size in bits.

7.4. Communication Overhead

We have investigated the impact of communication overhead by varying the number of messages containing digital signatures. Table 4 summarizes the calculated communication overhead for various signature schemes. We have chosen α within the finite field spectrum at 512 bits, ω at 160 bits, and p between 512 and 1024 bits for DSA. The lengths, as determined by the signature algorithm, are $|\omega| = |c| = 160$ bits and $|z| = |\mathcal{H}(M)| + |p| =$ 160 bits + 512 bits [8]. Thus, the total length of each message's signature is $|p| + |\omega| + 160 = 512 + 160 + 160 = 832$ bits. Similarly, the ECDSA signature is 832 bits. In our proposed LDSEGoV technique, the dimensions of c and z are each 160 bits. The signature size is $|\omega| + 160 = 160 + 160 = 320$ bits. The communication overhead was measured for

varying key sizes (160–384 bits). LDSEGoV and SCDSA generated compact signatures (320–1280 bits), reducing bandwidth usage by 60–92% compared to ECDSA and DSA. At 256-bit security, LDSEGoV's signature size was 9 times smaller than ECDSA and 16 times smaller than DSA. This grants LDSEGoV an equivalent 60% decrease in communication overhead per message as SCDSA, in comparison to DSA and ECDSA.

Table 4. Communication overhead comparison for digital signature schemes with operation counts and communication overhead represented in bits. (In this table, T_h, T_e, T_g, T_{sm}, and T_{key} denote the overhead of the hash function, exponentiation, generator function (g), ECC scalar multiplication and key generation, respectively).

Scheme	Key Gen.	Signature Gen.	Verification	Total Overhead	Total Cost (Bits)
DSA	T_{key}	$T_h + T_e + T_g + T_{sm}$	$T_h + T_e + T_g + T_{sm}$	$T_{key} + T_h + T_e + T_g + T_{sm}$	832
ECDSA	T_{key}	$T_h + T_e + T_g + T_{sm}$	$T_h + T_e + T_g + T_{sm}$	$T_{key} + T_h + T_e + T_g + T_{sm}$	832
Mughal et al. (SCDSA [8])	T_{key}	$T_h + T_e + T_g$	$T_h + T_e + T_g$	$T_{key} + T_h + T_e + T_g$	320
LDSEGoV (Proposed)	T_{key}	$T_h + T_e + T_g$	$T_h + T_e + T_g$	$T_{key} + T_h + T_e + T_g$	320

7.5. Computational Complexity and Comparative Analysis

7.5.1. Signing Phase Complexity

The cost of computation for generating the signature can be divided into four components: the computation of the hash function requires $O(|M|)$ steps for the message of length $|M|$; modular exponentiation α^{ε} requires $O(\log^2 \omega)$ operations; modular inversion $(c + x)^{-1}$ requires $O(\log^2 \omega)$ via the Extended Euclidean Algorithm; and modular multiplication requires $O(\log^2 \omega)$ operations. Hence, the total cost of generating the signature is $O(|M| + \log^2 \omega)$.

7.5.2. Verification Phase Complexity

Verification involves hash computation $O(|M|)$, complex exponentiation α^c requiring $O(\log^2 \omega)$, complex multiplication $Y \cdot \alpha^c$ requiring $O(\log^2 p)$, and complex exponentiation A^z requiring $O(\log^2 \omega)$. The total verification complexity is $O(|M| + \log^2 \omega + \log^2 p)$, where the dominant factor is complex arithmetic over $\mathbb{Z}_p[i]$ for production parameter sizes ($p \geq 2048$ bits).

7.5.3. Asymptotic Complexity Comparison

Table 5 compares the asymptotic computational complexity of LDSEGoV with DSA and ECDSA across key operations.

LDSEGoV achieves asymptotic complexity comparable to DSA and ECDSA, with significant performance enhancements in practice, such as being 6.5× faster in signing and 24.76× faster in verification. This is due to optimized constant factors in performing complex arithmetic $\mathbb{Z}_p[i]$ rather than asymptotic complexity.

Table 5. Asymptotic complexity comparison of digital signature schemes.

Scheme	Key Generation	Signing	Verification
LDSEGoV (Proposed)	$O(\log^2 p)$	$O(\|M\| + \log^2 \omega)$	$O(\|M\| + \log^2 \omega + \log^2 p)$
DSA	$O(\log^2 p)$	$O(\|M\| + \log^2 q)$	$O(\|M\| + \log^2 p)$
ECDSA	$O(\log n)$	$O(\|M\| + \log n)$	$O(\|M\| + \log n)$

7.5.4. Storage Requirements Analysis

Table 6 quantifies storage overhead for cryptographic key material and signatures across schemes at the 128-bit security level.

Table 6. Storage requirements comparison at the 128-bit security level.

Scheme	Private Key	Public Key	Signature	Total
LDSEGoV (Proposed)	256 bits	512 bits	320 bits	1088 bits
ECDSA	256 bits	512 bits	512 bits	1280 bits
DSA	256 bits	2048 bits	320 bits	2624 bits
RSA-2048	2048 bits	2048 bits	2048 bits	6144 bits

LDSEGoV achieves a 15% reduction in total storage compared to ECDSA, corresponding to a saving of 192 bits, and a 58% reduction compared to DSA, critical for resource-constrained CSC terminals and smart card applications where memory size is limited. Reducing ECDSA signature size from 512 to 320 bits directly reduces bandwidth in rural 2G/3G network deployments.

7.5.5. Qualitative Feature Comparison

Table 7 provides a qualitative assessment of security properties and implementation characteristics across signature schemes.

Table 7. Qualitative feature comparison of digital signature schemes.

Feature	LDSEGoV (Proposed)	SCDSA (Mughal et al. [8])	ECDSA	DSA
Formal Security Proof (EUF-CMA)	✓	×	✓	✓
Verification Consistency	✓	×	✓	✓
NIST-Compliant Hash Function	✓	×	✓	✓
Constant-Time Implementation Safe	✓	×	✓ *	✓ *
Compact Signature Size	✓	✓	×	×
Fast Verification (<1 ms)	✓	✓	×	×
Post-Quantum Resistant	×	×	×	×

✓ indicates the feature is supported; × indicates the feature is not supported. * Requires careful implementation to avoid timing leaks.

LDSEGoV combines formal security verification (not present in state-of-the-art works) with computational efficiency (superior to existing methods), making it a novel CDLP-based scheme ready for production in e-governance applications. The constant-time execution makes it more secure by preventing timing side-channel attacks, which are critical in government authentication services. However, all schemes are vulnerable to quantum attacks using Shor's algorithm and require post-quantum counterparts (ML-DSA, SLH-DSA) in the near future.

7.6. Limitations, Discussion, and Future Work

LDSEGoV provides 6.5× acceleration in signing, 24.76× in verification, and a 61% reduction in signature size compared to previous approaches. Moreover, LDSEGoV maintains its security as proven by formal validation using the EUF-CMA model with CDLP hardness. Nevertheless, similar to other DLP-based schemes such as RSA, DSA, and ECDSA, LDSEGoV is vulnerable to Shor's algorithm. In light of this, we recommend that government infrastructures use ML-DSA and SLH-DSA schemes standardized by NIST in FIPS 204 and FIPS 205, respectively. However, post-quantum schemes come at the cost of increased computational overhead and reduced performance. In terms of implementation, we recommend a three-step approach for government infrastructures. In the initial period, LDSEGoV is recommended for use in high-throughput authentication. In the transition period, a hybrid approach combining LDSEGoV and ML-DSA/SLH-DSA

using a dual-signature scheme is recommended. Finally, in a later period, we recommend a post-quantum approach.

Furthermore, the current evaluation emphasizes validation of algorithmic performance in a controlled laboratory environment. In addition to this, for deployment in a production environment within government infrastructures, further validation is necessary, including multi-platform validation on embedded devices and mobile processors, energy consumption profiling for battery-powered devices, integration with existing PKI infrastructures such as X.509 certificates and LDAP directory services, and certification for regulatory compliance such as Common Criteria and FIPS 140-3. The results presented show that LDSEGoV demonstrates a computational efficiency advantage. However, further assessment of production-readiness is necessary and has been identified as an area for further work. The claims regarding suitability for e-governance are based on computational capabilities, such as a single server capable of handling concurrent loads from hundreds of endpoints rather than a comprehensive assessment of production-readiness.

8. Conclusions and Outlook

This work presents LDSEGoV, a corrected and formally proven complex discrete logarithm-based digital signature scheme that addresses essential flaws identified in previous CDLP implementations through meticulous modular arithmetic, NIST-safe cryptographic primitives, and formal security proofs in the EUF-CMA model. Experimental results show that LDSEGoV provides a 6.5-fold improvement in signing performance and a 24.76-fold speedup in verification over ECDSA, with a 61% reduction in signature size for 128-bit security, making it suitable for e-governance authentication infrastructure. The results justify the implementation of cost-efficient authentication infrastructure for digital governance applications, especially in bandwidth-scarce networks.

LDSEGoV is best suited for operational authentication scenarios where signature validity is significant for years to decades. Long-term archival applications with a century-long lifespan (e.g., land records, constitutional texts) or post-quantum security requirements should use well-established RSA/ECDSA or NIST-standardized post-quantum cryptographic schemes (ML-DSA, SLH-DSA). Future research may explore the hardware design and implementation on resource-constrained platforms such as smart cards and IoT devices, hybrid cryptographic constructions for CDLP, and post-quantum primitives for transitory security, including formal verification of constant-time implementations to prevent side-channel attacks.

Author Contributions: Conceptualization, S.S. and P.S.; methodology, P.S.; software, O.P.; validation, S.S., P.S., and O.P.; formal analysis, P.S. and S.S.; investigation, S.S. and O.P.; resources, O.P.; data curation, S.S. and O.P.; writing—original draft preparation, P.S. and S.S.; writing—review and editing, S.S., P.S., and O.P.; visualization, P.S.; supervision, O.P.; project administration, O.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest. Although one author is affiliated with SafeQbit Technologies Private Limited, this affiliation did not influence the study design, data collection, analysis, or interpretation of results.

Appendix A

This appendix contains the complete Formal Security Proof of Theorem 1.

Game G_0: Standard EUF-CMA Challenge:

This game captures the standard notion of signature unforgeability. The challenger $\mathcal{C}$ performs the following:

1. Setup: Generate a random private key $x \xleftarrow{\$} [1, \omega - 1]$ with $\gcd(x, \omega) = 1$, compute public key $Y = \alpha^x \bmod p$, and give Y to the adversary $\mathcal{A}$.
2. Oracle Access: Provide $\mathcal{A}$ with two oracles:
   Signing Oracle $\mathcal{O}_{\text{Sign}}(M_i)$: On an input message M_i, the oracle generates a fresh random nonce ε_i, computes $c_i = \mathcal{H}(\alpha^{\varepsilon_i} \parallel M_i) \bmod \omega$, calculates $z_i = \varepsilon_i \cdot (c_i + x)^{-1} \bmod \omega$, and returns signature (c_i, z_i).
   Hash Oracle $\mathcal{H}(\cdot)$: On input $(u \parallel v \parallel M)$, returns a hash value. The oracle works consistently: repeated queries on the same input produce the same output.
3. Forgery: After polynomial-time running the queries to both Oracles, $\mathcal{A}$ outputs a forgery attempt (M^*, c^*, z^*) for a message M^* it has never queried to$\mathcal{O}_{\text{Sign}}$.
4. Winning Condition: $\mathcal{A}$ wins if (c^*, z^*) is a legitimate signature on M^* accepted by the regular verification algorithm.

Let ϵ_0 denote the probability that $\mathcal{A}$ wins in Game G_0.

Game G_1: Explicit Random Oracle Simulation

This construction changes the internal workings of the hash oracle without changing anything for the adversary. The challenger keeps a record, represented by the set $\mathcal{L}_H$ (which is empty initially), of all hash queries:

On query $(u \parallel v \parallel M)$:

1. If (u, v, M) exists in $\mathcal{L}_H$ with associated value h, return h.
2. Otherwise, sample $h \xleftarrow{\$} \mathbb{Z}_\omega$ uniformly, store $((u, v, M), h)$ in $\mathcal{L}_H$, and return h.

Indistinguishability: For $\mathcal{A}$, this change is purely syntactic. Both game scenarios respond identical to all queries. Therefore:

$$\Pr[\mathcal{A} \text{ wins in } G_1] = \Pr[\mathcal{A} \text{ wins in } G_0] = \epsilon_0$$

Game G_2: Forking Lemma Application

This game contains the forking method, which is the technical core of the security reduction. When $\mathcal{A}$ produces a valid forgery (M^*, c^*, z^*), the challenger rewinds $\mathcal{A}$ to an appropriate point and replays it with a different hash oracle response.

Forking Procedure:

1. Suppose $\mathcal{A}$ makes q_H hash queries $Q_1, Q_2, \ldots, Q_{q_H}$ before outputting forgery (M^*, c^*, z^*).
2. The challenger identifies the critical query $Q_j = (u^* \parallel v^* \parallel M^*)$ where u^*, v^* are derived from α^{ε^*} for some unknown ε^* used in the forgery.
3. Rewind $\mathcal{A}$ to just before query Q_j, and respond with a fresh random value $c'^* \neq c^*$ instead of the original c^*.
4. If $\mathcal{A}$ outputs a second valid forgery (M^*, c'^*, z'^*), we obtain two equations:

$$z^* = \varepsilon^* \cdot (c^* + x)^{-1} \bmod \omega \tag{A1}$$

$$z'^* = \varepsilon^* \cdot (c'^* + x)^{-1} \bmod \omega \tag{A2}$$

Key Extraction: Assuming $c^* \neq c'^*$ (which holds with probability $1 - 1/\omega \approx 1$), we can solve for the private key x:

From Equations (A1) and (A2), isolate ε^*:

$$\varepsilon^* = z^* \cdot (c^* + x) \bmod \omega \tag{A3}$$

$$\varepsilon^* = z'^* \cdot (c'^* + x) \bmod \omega \tag{A4}$$

Setting these equal and solving for x:

$$z^* \cdot (c^* + x) \equiv z'^* \cdot (c'^* + x) \pmod{\omega} \tag{A5}$$

$$z^* c^* + z^* x \equiv z'^* c'^* + z'^* x \pmod{\omega} \tag{A6}$$

$$z^* c^* - z'^* c'^* \equiv x(z'^* - z^*) \pmod{\omega} \tag{A7}$$

$$x \equiv \frac{z^* c^* - z'^* c'^*}{z'^* - z^*} \pmod{\omega} \tag{A8}$$

provided $z'^* \neq z^*$, which follows from $c'^* \neq c^*$.

Success Probability: Using the General Forking Lemma [29], if $\mathcal{A}$ succeeds with probability ϵ_0 after making q_H hash queries, then the rewinding algorithm produces two valid forgeries with probability:

$$\epsilon_2 \geq \epsilon_0 \left(\frac{\epsilon_0}{q_H} - \frac{1}{\omega} \right)$$

For any non-negligible ϵ_0 and polynomial q_H, this probability is non-negligible.

Game G_3: CDLP Reduction

We now construct a simulator $\mathcal{S}$ that solves CDLP using $\mathcal{A}$ as a subroutine. Given a CDLP challenge (α, Y) where $Y = \alpha^{\xi} \bmod p$ for unknown $\xi \in [1, \omega - 1]$, the simulator aims to compute ξ.

Simulation Strategy:

1. Embed Challenge: Set the public key $Y = Y$ and provide it to $\mathcal{A}$. The corresponding private key is $x = \xi$ (unknown to $\mathcal{S}$).
2. Answer Hash Queries: Maintain list $\mathcal{L}_H$ and respond to hash queries as in Game G_1 (uniformly random values).
3. Answer Signing Queries: For each signing query on message M_i:
   (a) Choose random $c_i, z_i \xleftarrow{\$} \mathbb{Z}_\omega$.
   (b) Compute $A_i = \alpha^{z_i^{-1}} \cdot Y^{-c_i \cdot z_i^{-1}} \bmod p$.
   (c) Extract (u_i, v_i) from A_i (real and imaginary parts).
   (d) Program the hash oracle: set $\mathcal{H}(u_i \parallel v_i \parallel M_i) = c_i$ by adding entry to $\mathcal{L}_H$.
   (e) Return signature (c_i, z_i).

   This is a valid signature because:

$$(Y \cdot \alpha^{c_i})^{z_i} = (\alpha^{\xi} \cdot \alpha^{c_i})^{z_i} \bmod p \tag{A9}$$

$$= \alpha^{(\xi + c_i) z_i} \bmod p \tag{A10}$$

$$= A_i \quad \text{(by construction)} \tag{A11}$$

4. Extract CDLP Solution: When $\mathcal{A}$ produces two forgeries (M^*, c^*, z^*) and (M^*, c'^*, z'^*) via the forking procedure in Game G_2, compute:

$$\xi = x = \frac{z^* c^* - z'^* c'^*}{z'^* - z^*} \bmod \omega$$

Verify: $Y \stackrel{?}{=} \alpha^{\xi} \bmod p$. Output ξ as the CDLP solution.

Indistinguishability: The simulated signatures (c_i, z_i) are distributed identically to real signatures because:

- In real scheme: $c_i = \mathcal{H}(\alpha^{\varepsilon_i} \parallel M_i)$ where ε_i is uniform $\Rightarrow$ c_i is uniform (by ROM).
- In simulation: c_i is directly sampled uniform.
- Both z_i values satisfy the verification equation.

Thus, $\mathcal{A}$ cannot distinguish the simulation from the real game, and succeeds with probability ϵ_2 (from Game G_2).

Final Reduction Analysis: We have shown that if an adversary $\mathcal{A}$ breaks LDSEGoV with advantage ϵ_0 after q_H hash queries, then there exists a CDLP solver $\mathcal{S}$ with success probability:

$$\epsilon_{\text{CDLP}} \geq \epsilon_0 \left(\frac{\epsilon_0}{q_H} - \frac{1}{\omega} \right)$$

Since CDLP is assumed computationally hard (no polynomial-time algorithm solves it with non-negligible probability), ϵ_{CDLP} must be negligible. This implies ϵ_0 is also negligible.

Concrete Security: For parameters $\omega \approx 2^{160}$ and adversary making $q_H \leq 2^{60}$ hash queries, breaking LDSEGoV with advantage $\epsilon_0 = 2^{-30}$ would yield a CDLP solver with advantage $\epsilon_{\text{CDLP}} \approx 2^{-90}$, contradicting the CDLP hardness assumption for 128-bit security.

Therefore, LDSEGoV is existentially unforgeable under adaptive chosen-message attacks in the Random Oracle Model, assuming CDLP hardness.

Interpretation for Practitioners: Theorem 1 guarantees that even if the adversary views a polynomial number of valid signatures and can choose adaptively which messages to sign (as in the case of chosen-plaintext attacks), it is impossible to sign a new message without breaking the underlying CDLP. The assumption made in the Random Oracle Model (ROM) is common practice when using cryptographic hash functions such as SHA-256, which currently have no known structural weaknesses relevant to this construction.

References

1. Fernández-Caramés, T.M. From Pre-Quantum to Post-Quantum IoT Security: A Survey on Quantum-Resistant Cryptosystems for the Internet of Things. *IEEE Internet Things J.* **2020**, *7*, 6457–6480. [CrossRef]
2. Galbraith, S.D. *Mathematics of Public Key Cryptography*; Cambridge University Press: Cambridge, UK, 2012.
3. Barker, E. *Recommendation for Key Management: Part 1—General*; Technical Report NIST SP 800-57 Part 1 Rev. 5; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2020. [CrossRef]
4. Boudot, F.; Gaudry, P.; Guillevic, A.; Heninger, N.; Thomé, E.; Zimmermann, P. *Comparing the Difficulty of Factorization and Discrete Logarithm: A 240-Digit Experiment*; Springer: Cham, Switzerland, 2020. [CrossRef]
5. Kamal, A.; El-Kamchochi, H.A.; El-Fahar, A.; Hagras, E.A.A. Conic curve encryption and digital signature based on complex number theory for cybersecurity applications. *Sci. Rep.* **2025**, *15*, 21566. [CrossRef] [PubMed]
6. Hla, N.N.; Aung, T.M. Intelligent Computing on Complex Numbers for Cryptographic Applications. In *Handbook of Intelligent Computing and Optimization for Sustainable Development*; John Wiley & Sons, Ltd.: Hoboken, NJ, USA, 2022; Chapter 4, pp. 69–79 . [CrossRef]
7. Aung, T.M.; Hla, N.N. A complex number approach to elliptic curve cryptosystems over finite fields: Implementations and experiments. In *Proceedings of the 2019 International Conference on Computer Communication and Informatics (ICCCI)*; IEEE: New York, NY, USA, 2019; pp. 1–8. [CrossRef]
8. 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]
9. Stevens, M.; Bursztein, E.; Karpman, P.; Albertini, A.; Markov, Y. The First Collision for Full SHA-1. In *Proceedings of the Advances in Cryptology—CRYPTO 2017*; Katz, J., Shacham, H., Eds.; Springer: Cham, Switzerland, 2017; pp. 570–596.
10. Leurent, G.; Peyrin, T. SHA-1 is a Shambles: First Chosen-Prefix Collision on SHA-1 and Application to the PGP Web of Trust. In *Proceedings of the 29th USENIX Security Symposium (USENIX Security 20)*; USENIX Association: Berkeley, CA, USA, 2020; pp. 1839–1856.
11. Katz, J.; Lindell, Y. *Introduction to Modern Cryptography: Principles and Protocols*; Chapman and hall/CRC: New York, NY, USA, 2007. [CrossRef]
12. Peng, S.; Zhao, L.; Al-Dubai, A.Y.; Zomaya, A.Y.; Hu, J.; Min, G.; Wang, Q. Secure Lightweight Stream Data Outsourcing for Internet of Things. *IEEE Internet Things J.* **2021**, *8*, 10815–10829. [CrossRef]
13. Sharif, A.; Ginting, D.S.; Dias, A.D. Securing the Integrity of PDF Files using RSA Digital Signature and SHA-3 Hash Function. In *Proceedings of the 2021 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA)*; IEEE: New York, NY, USA, 2021; pp. 154–159. [CrossRef]

14. Aufa, F.J.; Endroyono; Affandi, A. Security System Analysis in Combination Method: RSA Encryption and Digital-Signature Algorithm. In *Proceedings of the 2018 4th International Conference on Science and Technology (ICST)*; IEEE: New York, NY, USA, 2018; pp. 1–5. [CrossRef]
15. Meshram, C.; Ibrahim, R.W.; Obaid, A.J.; Meshram, S.G.; Meshram, A.; El-Latif, A.M.A. Fractional chaotic maps based short signature scheme under human-centered IoT environments. *J. Adv. Res.* **2021**, *32*, 139–148. [CrossRef] [PubMed]
16. Ahmed, A.A.; Barukab, O.M. Unforgeable Digital Signature Integrated into Lightweight Encryption Based on Effective ECDH for Cybersecurity Mechanism in Internet of Things. *Processes* **2022**, *10*, 2631. [CrossRef]
17. Yavuz, A.A.; Darzi, S.; Nouma, S.E. Lightweight and Scalable Post-Quantum Authentication for Medical Internet of Things. *arXiv* **2023**, arXiv:2311.18674.
18. 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. [CrossRef]
19. Sarkar, A.; Guha Roy, D.; Datta, P. An Overview of the Discrete-Logarithm Problem in Cryptography. In *Proceedings of the Third International Conference on Advanced Computing and Applications*; Giri, D., Das, S., Corchado Rodríguez, J.M., De, D., Eds.; Springer: Singapore, 2024; pp. 129–143.
20. Battarbee, C.; Darbinyan, A.; Kahrobaei, D. *Hard Instances of Discrete-Logarithm Problem and Cryptographic Applications*; Cryptology ePrint Archive; Paper 2025/1461; 2025. Available online: https://eprint.iacr.org/2025/1461 (accessed on 12 February 2026).
21. Barbulescu, R. *Cryptanalysis of Factoring and the Discrete-Logarithm Problem and Their Ramifications on the Smooth Numbers and Modular Curves*; Accreditation to Supervise Research; Université de Bordeaux: Talence, France, 2025.
22. Wang, X.; Yin, Y.L.; Yu, H. Finding collisions in the full SHA-1. In *Proceedings of the Annual International Cryptology Conference*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 17–36.
23. Jancar, J.; Sedlacek, V.; Svenda, P.; Sys, M. Minerva: The Curse of ECDSA Nonces. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2020**, *4*, 281–308. [CrossRef]
24. Macchetti, M. *A Novel Related Nonce Attack for ECDSA*; Cryptology ePrint Archive; Paper 2023/305; 2023. Available online: https://eprint.iacr.org/2023/305 (accessed on 12 February 2026).
25. Cao, W.; Shi, H.; Chen, H.; Wei, W.; Chen, J. Lattice-Based Weak Curve Fault Attack on ECDSA. In *Proceedings of the IFIP Advances in Information and Communication Technology*; Jøsang, A., Futcher, L., Hagen, J., Eds.; ICT Systems Security and Privacy Protection; Springer: Oslo, Norway, 2021; Volume AICT-625 , pp. 146–161. [CrossRef]
26. Hou, X.; Breier, J.; Jap, D.; Ma, L.; Bhasin, S.; Liu, Y. Physical security of deep learning on edge devices: Comprehensive evaluation of fault injection attack vectors. *Microelectron. Reliab.* **2021**, *120*, 114116. [CrossRef]
27. Blessing, J.; Specter, M.A.; Weitzner, D.J. Cryptography in the Wild: An Empirical Analysis of Vulnerabilities in Cryptographic Libraries. In *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*; ASIA CCS '24; Association for Computing Machinery: New York, NY, USA, 2024; pp. 605–620. [CrossRef]
28. Geimer, A.; Vergnolle, M.; Recoules, F.; Daniel, L.A.; Bardin, S.; Maurice, C. A Systematic Evaluation of Automated Tools for Side-Channel Vulnerabilities Detection in Cryptographic Libraries. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*; CCS '23; Association for Computing Machinery: New York, NY, USA, 2023; pp. 1690–1704. [CrossRef]
29. Pointcheval, D.; Stern, J. Security Arguments for Digital Signatures and Blind Signatures. *J. Cryptol.* **2000**, *13*, 361–396. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.