Algebraic Anomalies in ECDSA Signatures Enabling Private Key Recovery Under Ideal Random Nonces

Andriy Polishchuk
CRYPTON Systems Lab
andriy.polishchuk.a@gmail.com
linkedin.com/in/andriy-polishchuk

May 29, 2025

Abstract

This paper presents three novel vulnerabilities in the Elliptic Curve Digital Signature Algorithm (ECDSA), discovered through mathematical analysis and empirical exploration. Each vulnerability enables specific attack vectors that compromise the algorithm's reliability under certain assumptions. We provide theoretical foundations, demonstrate practical examples based on a test elliptic curve, and discuss implications for digital signature security.

Keywords: ECDSA, secp256k1, Cryptographic vulnerabilities, Elliptic curves, ECC

Author's Note

This paper presents the first formal publication of a foundational discovery made independently in 2022, concerning algebraic vulnerabilities in ECDSA transactions. That investigation revealed structural patterns within valid signatures that may allow classification and potential exploitation under ideal conditions.

A second, separate line of ongoing research explores the possibility of fully inverting the ECDSA process by reconstructing the private key solely from the public key, based on the hypothesis that a unique algebraic path of inversion may exist on elliptic curves.

The two investigations are conceptually related but methodologically independent. This paper focuses exclusively on the first, algebraic anomalies in the transaction structure.

Warning

This research does not pose an immediate threat to the security of real-world cryptographic systems. All examples and vulnerability classifications were derived using synthetic data on a test elliptic curve with significantly reduced parameters. No cryptographic implementations, assets, or networks were attacked or compromised during this study.

This work aims to expose an overlooked algebraic property of ECDSA that may be of theoretical importance and interest to the cryptographic research community. It is not intended to serve as a practical attack but rather to stimulate further study and reinforce the need for rigorous analysis of even well-established cryptographic primitives.

All data, computations, and proofs were produced in controlled test environments, and the results should be interpreted in that context.

1 Introduction

The Elliptic Curve Digital Signature Algorithm (ECDSA) is a widely adopted cryptographic scheme to ensure the authenticity and integrity of digital data. It is particularly prominent in blockchain technologies, including Bitcoin and Ethereum, where it underpins the security of transactions and wallet ownership. ECDSA combines elliptic curve mathematics with digital signature principles for efficient and secure authentication.

Despite its strong theoretical foundation, ECDSA's security relies on several critical assumptions, such as the uniqueness of the private key given a public key and the unpredictability of nonces used during signature generation. Any deviation from these assumptions—whether due to implementation flaws, randomness issues, or hidden algebraic patterns—can lead to serious vulnerabilities.

This paper introduces and analyses three algebraic anomalies in ECDSA signatures, each revealing structural weaknesses that may allow private key recovery even under ideal conditions. Unlike previous attacks that rely on biased or reused nonces, our findings demonstrate that vulnerabilities can arise from the intrinsic mathematical properties of ECDSA itself. These results challenge long-standing beliefs about ECDSA's invulnerability to attacks under perfect randomness and lay the groundwork for a deeper understanding of its algebraic structure.

2 ECDSA Background

ECDSA operates over a set of mathematical operations defined on an elliptic curve E over a finite field $\mathbb{F}_p$. The equation describes the curve:

$$E : y^2 = x^3 + ax + b \mod p$$

where p is a large prime, and a, b are curve parameters satisfying $4a^3 + 27b^2 \neq 0$ to ensure non-singularity. A base point G of prime order n is also defined on the curve.

Key Generation

- The private key x is randomly chosen from $[1, n-1]$
- The public key is $P = xG$

Signature Generation

To sign a message m:

1. Hash the message: $z = H(m)$
2. Choose a random nonce $k \in [1, n-1]$
3. Compute the point $R = kG = (r, y_r)$, and let $r = x$-coordinate $\mod n$
4. Compute the signature component:

$$s = k^{-1}(z + xr) \mod n$$

5. The signature is the pair (r, s)

Signature Verification

Given a public key P, message m, and signature (r, s):

1. Hash the message: $z = H(m)$
2. Compute:
$$u_1 = zs^{-1} \mod n, \quad u_2 = rs^{-1} \mod n$$
3. Compute:
$$R = u_1 G + u_2 P$$
4. The signature is valid if $r \equiv x_R \mod n$

This process assumes k is secret and truly random. However, as demonstrated in this paper, even under ideal conditions, where all parameters are generated correctly, hidden algebraic patterns may be exploitable by an adversary.

3 Vulnerability detection mechanism

3.1 Theory

Let the standard ECDSA signature be defined as:

$$s = k^{-1}(z + xr) \mod n$$

Multiplying both sides by k:

$$sk = z + xr \mod n$$

Let:

$$s_{zk} = zk \mod n, \quad s_{rxk} = rxk \mod n$$

Then:

$$s = s_{zk} + s_{rxk} \mod n$$

Also define:

$$s_{zr} = zr \mod n$$

$$q = s_{zr} - s \quad (\text{only if } s_{zr} > s)$$

$$a = s \mod q$$

Now compute the fractions:

$$m_1 = \frac{s_{zk}}{a}, \quad m_2 = \frac{s + s_{zr}}{s_{rxk}}$$

A transaction is classified as vulnerable when the following conditions are met:

- $s_{zk} > 0$, $s_{rxk} > 0$
- $s = s_{zk} + s_{rxk}$
- $s_{zr} > s$
- $a > 1, a \neq s, a = s_{zr} \mod q$

Based on values of m_1 and m_2, we distinguish:

- **Case A:** denominator$(m_1) = 1 \wedge m_1 \neq m_2 \wedge m_1 = 1$
- **Case B:** denominator$(m_1) = 1 \wedge m_1 = m_2 \wedge m_1 > 1$

These formulas were derived empirically by scanning large volumes of valid ECDSA signatures generated with ideal random nonces k. Despite the lack of nonce reuse or implementation flaws, we show that certain patterns emerge algebraically—offering an attack surface for classification and analysis.

3.2 Example

All transactions analyzed in this paper were generated using a test elliptic curve with the following parameters:

- Field characteristic: $p = 10007$
- Curve coefficients: $a = 48$, $b = 22$
- Base point: $G = (4,\ 1668)$
- Subgroup order: $n = 9967$

This simplified elliptic curve was used in place of secp256k1 to allow efficient large-scale experimentation. Over **600 million** ECDSA signatures were generated and tested on this curve under ideal conditions (unique private key, uniformly random nonce). The vulnerabilities presented in this study were identified and validated within this high-volume synthetic dataset. Given that the signature function depends on three independent parameters— the private key x, the message hash z, and the nonce k—all uniformly drawn from the set $\{1, 2, \ldots, n-1\}$, the theoretical space of unique transaction instances is:

$$(x, z, k) \in \{1, \ldots, n-1\}^3 \quad \Rightarrow \quad \text{Total space size} = (n-1)^3$$

For the experimental curve with $n = 9967$, the total number of possible unique ECDSA transaction instances equals:

$$(9967 - 1)^3 = 9966^3 \approx 9.89 \times 10^{11}$$

In this study, we generated and analyzed a representative sample of 600 000 000 transaction instances, which corresponds to approximately:

$$\frac{6 \times 10^8}{(n-1)^3} \approx 0.06\%$$

of the full theoretical transaction space. Below are 10 transactions (Tx) for all three vulnerability types (cases A, B), which were taken from a total transaction set of 600 million.

Tx	Case	s	s_{zk}	s_{rxk}	s_{zr}	z	r	x	k	q	a	m_1	m_2
1	A	7584	1204	6380	8860	9572	4141	3166	5094	1276	1204	1	-
2	A	6417	1413	5004	8919	5674	6162	7762	8107	2502	1413	1	-
3	A	4898	1242	3656	6726	9047	9288	5053	497	1828	1242	1	-
4	A	8609	769	7840	9589	3512	5761	4473	7291	980	769	1	-
5	A	7106	170	6936	9418	2976	251	4955	9036	2312	170	1	-
6	B	4559	1940	2619	5917	142	533	8937	8998	1358	485	4	4
7	B	7074	3930	3144	8646	2789	71	1351	7281	1572	786	5	5
8	B	6831	5616	1215	8964	6859	3576	4092	5963	2133	432	13	13
9	B	5015	885	4130	7375	7959	41	1149	2397	2360	295	3	3
10	B	1316	560	756	1708	1007	2585	4306	9344	392	140	4	4

4 Vulnerability A: When $m_1 = 1$ is satisfied

4.1 Theory

(see Case A of "Vulnerability detection mechanism")

Let a standard ECDSA signature be defined as:

$$s = k^{-1}(z + r \cdot x) \mod n$$

Multiplying both sides by k:

$$s \cdot k = z + r \cdot x \mod n$$

We define:

$$s_{zk} = z \cdot k \mod n, \quad s_{rxk} = r \cdot x \cdot k \mod n$$

$$s = s_{zk} + s_{rxk} \mod n$$

Suppose the first component s_{zk} can be expressed as:

$$s_{zk} = a \cdot m_1$$

where $a = s \mod q$ and $q = z \cdot r - s$, then:

1. Compute the nonce:

$$k = s_{zk} \cdot z^{-1} \mod n$$

2. Extract the second part of the signature:

$$s_{rxk} = s - s_{zk}$$

3. Recover the private key:

$$x = s_{rxk} \cdot (r \cdot k)^{-1} \mod n$$

While m_1 can be a small integer greater than one in many practical cases, this study specifically filters for transactions where $m_1 = 1$. In such cases, the value of s_{zk} is directly equal to a, and the private key can be recovered almost instantly using only public parameters. This makes the attack not only computationally trivial but effectively instantaneous in these identified cases.

4.2 Example

We now examine five example transactions of Case A:

Tx	s	s_{zk}	s_{rxk}	z	z^{-1}	r	k	$r \cdot k$	$(r \cdot k)^{-1}$	x
1	7584	1204	5094	9572	1842	4141	6380	4082	691	3166
2	6417	1413	8107	5674	8040	6162	5004	730	3782	7762
3	4898	1242	497	9047	5016	9288	3656	1415	7234	5053
4	8609	769	7291	3512	6814	5761	7840	2513	2697	4473
5	7106	170	9036	2976	1343	251	6936	5527	2925	4955

Each of these transactions satisfies the condition $s = s_{zk} + s_{rxk}$, and was specifically selected such that $m_1 = 1$, meaning $s_{zk} = a$. This condition allows for the immediate computation of the nonce k using the public hash value z as $k = s_{zk} \cdot z^{-1} \mod n$. Once k is known, the private key x can be directly recovered using public values via $x = s_{rxk} \cdot (r \cdot k)^{-1} \mod n$.

This subset demonstrates the most severe variant of the vulnerability: private key recovery that requires no iteration over m_1, making the attack effectively instantaneous whenever such a transaction occurs.

5 Vulnerability B: When $m_1 = m_2 > 1$ is satisfied

5.1 Theory

(see Case B of "Vulnerability detection mechanism")

Let a standard ECDSA signature be defined as:

$$s = k^{-1}(z + r \cdot x) \mod n$$

Multiplying both sides by k yields:

$$s \cdot k = z + r \cdot x \mod n$$

Let us define:

$$s_{zk} = z \cdot k \mod n, \quad s_{rxk} = r \cdot x \cdot k \mod n$$

$$s = s_{zk} + s_{rxk} \mod n$$

We also define the auxiliary public value:

$$s_{zr} = z \cdot r \mod n$$

In this case, we are interested in signatures where the following equality is satisfied:

$$m_1 = \frac{s_{zk}}{a} = \frac{s + s_{zr}}{s_{rxk}} = m_2, \quad \text{and} \quad m_1 > 1$$

This condition leads to a direct analytical method for recovering m_1 from public values using a quadratic equation.

We define the discriminant:

$$D = s^2 - 4a(s_{zr} + s)$$

Then compute the candidate solution:

$$m_1 = \frac{s \pm \sqrt{D}}{2a}$$

In cases of Vulnerability B, one of these roots is a valid integer. Once m_1 is recovered, the following procedure is used:

1. Compute $s_{zk} = a \cdot m_1$
2. Compute the nonce: $k = s_{zk} \cdot z^{-1} \mod n$
3. Compute $s_{rxk} = s - s_{zk}$
4. Recover the private key: $x = s_{rxk} \cdot (r \cdot k)^{-1} \mod n$

This results in full key recovery in constant time, based entirely on deterministic and public parameters.

5.2 Example

We now examine five example transactions of Case B:

Tx	s	s_{zk}	s_{rxk}	z	z^{-1}	r	k	$r \cdot k$	$(r \cdot k)^{-1}$	x	m_1
1	4559	1940	2619	142	1474	533	8998	1807	3497	8937	4
2	7074	3930	3144	2789	679	71	7281	8634	8823	1351	5
3	6831	5616	1215	6859	9874	3576	5963	4275	9552	4092	13
4	5015	885	4130	7959	273	41	2397	8574	7706	1149	3
5	1316	560	756	1007	871	2585	9344	4199	8417	4306	4

The following five transactions were identified as instances of Vulnerability B, in which the multipliers m_1 and m_2 are equal and greater than one. This structural coincidence allows for the analytical derivation of m_1 using the quadratic formula, enabling full recovery of the private key with no brute-force iteration. All values shown below are derived from public components.

6 Implications for Security

The vulnerabilities identified in this paper demonstrate that, under specific algebraic conditions, private key recovery in ECDSA is theoretically possible without requiring leakage of the nonce or failures in random number generation. This reveals a new class of risks based on the algebraic structure of the signature equation itself.

In particular, Vulnerability A exposes cases where $m_1 = 1$ results in direct recovery of the key, and Vulnerability B covers cases where $m_1 = m_2 > 1$, enabling recovery via closed-form expressions. These discoveries underscore the importance of analyzing the structure of the ECDSA signature space beyond conventional attack surfaces.

Important note on generalization. All experimental results in this paper were derived using a simplified test elliptic curve with parameters:

$$p = 10007, \quad a = 48, \quad b = 22, \quad n = 9967, \quad G = (4,\ 1668)$$

Although this test curve differs from secp256k1, it preserves the same algebraic logic underlying ECDSA. The presence of algebraic anomalies on this curve proves the existence of deterministically invertible signature structures under ideal randomness.

The actual frequency and distribution of such anomalies on secp256k1 remain unknown, but the probability is not provably zero. Given the massive size of the key space $(n-1)^3$, the appearance of rare but structurally vulnerable signatures cannot be ruled out.

This research does not claim to break Bitcoin or ECDSA broadly. Rather, it provides a foundation for future work in anomaly detection, signature validation frameworks, and deeper cryptanalytic modelling.

7 Conclusion

This paper introduced two classes of algebraic anomalies in ECDSA signatures—Vulnerability A and Vulnerability B—that allow for deterministic private key recovery using only public information. These cases demonstrate that even when nonces are truly random, some signatures exhibit structural patterns that render them cryptographically weak.

While our experiments were limited to a small test elliptic curve, the consistency of the algebraic behavior suggests these phenomena merit further exploration on real-world curves like secp256k1.

We hope this work stimulates further investigation into the statistical structure of ECDSA signatures, as well as the development of tools to identify and assess such anomalies in blockchain environments.

Acknowledgments

The author gratefully acknowledges the open-source Python and SymPy communities for the foundational tools that enabled large-scale elliptic curve simulations.

This research was conducted independently, without institutional funding or affiliation. Special thanks to the broader cryptography community for inspiring deeper exploration into ECDSA signature structures.

The feedback of AI-assisted tooling (ChatGPT) was used extensively for formalization, LaTeX formatting, and iterative verification throughout the writing process.

References

[1] D. Johnson, A. Menezes, and S. Vanstone.*The Elliptic Curve Digital Signature Algorithm (ECDSA).* 2001.

[2] S. Nakamoto.*Bitcoin: A Peer-to-Peer Electronic Cash System.* 2008.

[3] L. Washington.*Elliptic Curves: Number Theory and Cryptography.* CRC Press, 2003.

Supplementary Materials

All scripts, datasets, and computational notebooks used in this research are publicly available at:

`https://github.com/apolish/ECDSA-Anomalies`

This repository includes:

- Python scripts for generating and categorizing over 500 million test transactions
- Datasets of ECDSA signatures exhibiting algebraic anomalies (Case A and B)
- Worked examples and step-by-step verification
- Final PDF and LaTeX source of this paper